KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Science

A study and implementation of robust estimators for multivariate and functional data.

Kaveh Vakili

Supervisor:
Prof. dr. P. Rousseeuw

Dissertation presented in partial fulfillment of the requirements for the PhD in Science(Phd): Statistics

June 2016

A study and implementation of robust estimators for multivariate and functional data.

Kaveh VAKILI

Examination committee:
Prof. dr. S. Vaes, chair
Prof. dr. P. Rousseeuw, supervisor
Prof. dr. J. Beirlant
Prof. dr. C. Croux
Prof. dr. T. Verdonck
Prof. dr. I. Van Keilegom
(U.C.Louvain)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Science(Phd): Statistics

June 2016

Preface

出る釘は打たれる

Japanese proverb.

TE OCCIDERE POSSUNT SED
TE EDERE NON POSSUNT NEFAS EST

Moto of the Enfield Tennis Academy.

I would like to thank my promoter for helping me start this thesis as well two mediators, two members of the supervisory committee and the head of the doctoral school for helping me finish it.

I am also indebted to my promoter for his contribution to robust statistics.

Abstract

[Beran, 2003] defined statistics as the study of algorithms for data analysis. In many situations several variables need to be taken into account simultaneously to accurately describe the patterns in the data. In practice this is done by fitting a model to the data. Often, real life data sets also contain outliers, i.e. observations inconsistent with the multivariate patterns of the majority of the data. Outliers tend to exert a disproportionate pull on the fit thereby blurring the main patterns in the data as well as their true outlyingness. Robust estimators are designed to prevents arbitrarily outliers from exerting undue influence on the fitted model.

Several approaches to obtain robust estimates exist depending on the characteristics of the data. In Chapter two we compare many robust estimators of multivariate location and scatter by means of simulations. An important conclusion of this chapter is that in many situations those algorithms do not always reveal the outliers (or prevent them from swaying the fit).

A fundamental problem in statistics is that of estimating the linear relationship between multiple design variables and a single response variable. Though many robust algorithms have been developed to fit such (so-called regression) models, sometimes, in applications, they suffer from being too slow and/or random. In Chapter 3, we propose a new robust algorithm for regression that is both quick and deterministic.

Functional data can be seen as structured multivariate data collected from an underlying collection of smooth processes, or "curves". Often, the interest will be on estimating the main features of these curves such as their central tendency or variability as well as identifying sub-groups of curves detached from the majority. Chapter 4 offers an new approach to do this and compares it with state of the art alternatives by means of simulations.

The general availability of open source, portable, versatile and easy to use software libraries written in an up to date programing paradigm is an important

component in the success of procedures in computational statistics. A library of algorithms for depth and related data analytical tool has been developed as part of the work that led to this thesis. Chapter 5 show cases many components of this library and discuss how these improve upon available alternatives.

Beknopte samenvatting

[Beran, 2003] definieert statistiek als de studie van algoritmes voor data analyses. In veel gevallen moeten verschillende variabelen tegelijkertijd in beschouwing genomen worden om de patronen in de data nauwkeurig te beschrijven. In de praktijk wordt dit gedaan aan de hand van een statistisch model. In de praktijk bevatten datasets ook uitschieters, i.e. observaties die inconsistent zijn met de multivariate patronen in de meerderheid van de data. Uitschieters hebben de neiging om een disproportionele invloed uit te oefenen op de schattingen. Indien dit fenomeen niet gecorrigeerd wordt, zullen ze de belangrijkste relaties in de data vertroebelen, zowel als hun 'outlyingness'. Robuuste schatters worden ontworpen zodat ze vermijden dat arbitraire uitschieters ongewenste invloeg uitoefenen op het geschatte model.

Verschillende benaderingen tot het bekomen van rubuuste inschattingen bestaan naargelang de karakteristieken van de data. In hoofdstuk twee vergelijken we verschillende robuuste schatter (van multivariate locatie en spreide) door middel van simulaties. Een belangrijke conclusie van dit hoofdstuk is dat deze algoritmes uitschieters niet altijd op betrouwbare wijze onthullen (of voorkomen dat ze het geschatte model vertekenen).

Een fundamenteel probleem in de statistiek is het inschatten van de lineaire relatie tussen meerdere verklaarde veranderlijke en één afzonderlijke afhankelijke variabele. Hoewel er vele rubuuste algoritmes ontwikkeld zijn om in zulke (zogenaamde regressie-) modellen te passen, lijden deze soms in de praktijk onder hun traagheid en/of willekeurigheid. In hoofdstuk 3, suggereren we een nieuw robuust algoritme voor regressie dat zowel snel als deterministisch is.

Functionele data kan gezien worden als gestructureerde multivariate data die verzameld werd uit een onderliggende verzameling van zuivere processen, of "curves". Vaak zal onze focus liggen bij het inschatten van de voornaamste eigenschappen van deze curves, zoals hun neiging om of hun variabiliteit, zowel als het identificeren van subgroepen van curves die losgekoppeld zijn van de

meerderheid. Hoofdstuk 4 beschrijft een nieuwe aanpak om dit te doen en vergelijkt het met courante alternatieven door middel van simulaties.

De algemene beschikbaarheid van open source, draagbare, versatiele en toegangkelijke software bibliotheken, geschreven volgens een up-to-date programmeringsparadigma, is een belangrijke factor in het succes van procedures in de computationele statistiek. Een bibliotheek van algoritmes voor diepte en gerelateerde data analytisch instrument zijn ontwikkeld tijdens het werk dat aan de basis ligt van de totstandkoming van deze thesis. Hoofdstuk vijf toont veel componenten van deze bibliotheek en bespreekt hoe deze een vooruitgang betekenen op de momenteel beschikbare alternatieven.

Contents

Chapter 1

Introduction

1.1 General statement of the problem

In the eighteenth century, the word "robust" was used to refer to someone who was strong, yet boisterous, crude, and vulgar.[Stigler, 1973]

Robust statistics, like all branches of statistics, is concerned with the related problems of finding patterns in the data and quantifying the reliability of those patterns. This is done by fitting (or adjusting) statistical models to the data. Robust statistics, however, distinguishes itself from the other branches of statistics most notably in that it is not predicated on a hopeful view of the data.

Generally speaking, robust methods for data analysis are designed under the assumption that they will be handed unfriendly data-sets and used with the expectation that they will be able to cope with them. More specifically, the problem can be stated as follows. We are being given a potentially *contaminated* data set, e.g. one in which an unknown fraction of the original observations have been replaced by outliers, data points about which nothing can be safely assumed. For all intents and purposes, these outliers can be thought of as having been set up by an adversary with full knowledge of our methods and bent on hiding the true patterns in the data from us. Naturally, our objective is to defeat this adversary by finding a fit as close as possible to the one we would have found had we use the original, uncontaminated, data.

From a practical point of view, outliers are observations that are inconsistent with the pattern of the majority of the data. Outliers can have a variety

of causes such as unaccounted heterogeneity, bad data imputation, technical failure or fraud. If left unchecked, they influence the estimated parameters by disproportionately pulling the fitted model towards themselves. In this way, outliers obscure the main relationships in the data and their true outlyingness. In practice, we want to find the outliers to bound their influence on the fit and to study them as objects of interest in their own right. Furthermore, detecting outliers in settings involving more than two variables is difficult because we can not inspect the data visually and have to rely on algorithms instead.

Robust fitting procedures are important because real life data contain outliers and classical methods are, often to a surprising extent for the non initiated, sensitive to them. Because the effects of outliers is most manifest when the number of variable is two (so that the data can be vizualized), I will illustrate this by means of two popular bivariate examples. The first example uses the data for the Hertzsprung-Russell Diagram of the Star Cluster CYG OB1, of 47 stars in the direction of Cygnus [Rousseeuw and Leroy, 1987] and shown in the left plot of Figure 1.1. The second example uses the engine exhaust data set. This data set presents the results of an experiment in which ethanol was burned in a single cylinder automobile test engine and the resulting exhaust gazes where analyzed for their nitric oxide nitrogen dioxide content [Brinkman, 1981], shown in the right plot of Figure 1.1.

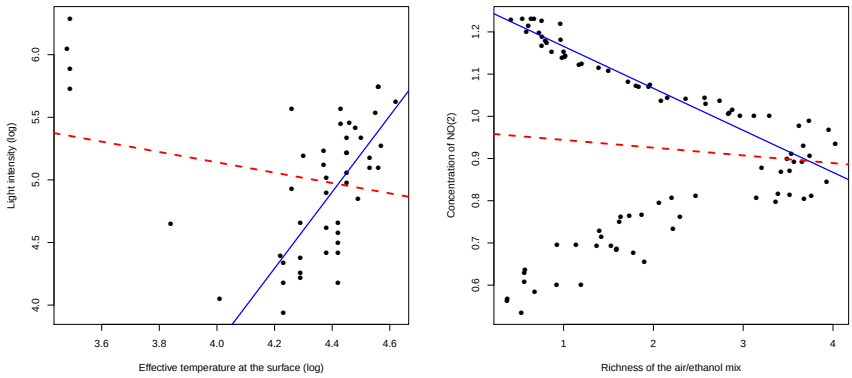


Figure 1.1: Left:Hertzsprung-Russell Diagram of the Star Cluster CYG OB1: Logarithm of the star's effective temperature at the surface versus the logarithm of its light intensity. Right: Motor exhaust data-set: richness of the air-ethanol mix versus concentration of NO(2).

The left plot in Figure 1.1 illustrate how the four so called giant stars in the upper left corner of the plot exert an out-sized influence on the classical OLS

fit (shown here as the dashed line). Despite being less than 10% of the sample, their collective pull on the fit found by OLS is so large that they now appear consistent with the fitted model and their residuals with respect to it does not reveal them. In contrast, a robust fit such as FastLTS (shown here as a full line) is not unduly attracted by the outliers. Consequently, the residuals with respect to the FastLTS fit can be used to reveal the outliers. The right plot in Figure 1.1 illustrate again how the OLS fit (shown as a dashed line) tries to accommodate all the observations, thereby producing a poor fit of data (in the sense of [Davies, 1995], say) and preventing the residuals from clearly revealing the outliers. In contrast, the FastLTS fit (shown here as a full line) is not unduly attracted to the members of the minority components in the data. Consequently, the fit found by FastLTS not only fits the main relationship in the data better, the residuals from it also reveal the true outliers.

This thesis is a work in applied mathematics. The focus is on solving problems concocted by day to day experience and the objective is to come up with solutions good enough to be useful in a given a practical setting: e.g. for a given set of purposes and means available to achieve them. As usual, the appeal of the challenge comes from the tension between the conflicting requirements imposed by the objectives to attain and the means available to attain them and every robust data analysis method will have to strike a compromise between these two aspects. In this thesis, I will explore some of these possible compromises and their practical consequences.

This thesis is about methods for robust estimation and anomaly detection in the context of multivariate and functional data. Multivariate data are datasets where several potentially dependent measurements are recorded each on a larger number of observations which we believe are independent of one another. Functional data are collection of naturally structured multivariate datasets which, though we observe them only at finitely many occasions, we believe are generated by underlying smooth functions. As an example of the former type of dataset consider $p > 1$ morphological measurements taken on $n > p$ individuals at a given time. As an example of the second type of dataset consider $p > 1$ morphological measurements taken on $n > p$ individuals at T different time periods. In both cases, the common thread is that the number of measurements is typically larger than three so that we cannot visualize the data and have to rely on algorithms (and models) instead [Rousseeuw and Van Zomeren, 1990].

In this thesis, I will be interested in particular to those situations where one qualifies the general problem of fitting statistical models robustly with a simplifying postulate. All the solutions studied in this thesis bound the adversary to play by the rules of the so-called Tukey-Huber contamination model [Tukey, 1962]. Under this postulate, a large proportion $(1 - \varepsilon)$ of the observations are well approximated as draws from a member of a classical,

simple and well behaving, family of distribution. The remaining observations can be affected by the adversary in unspecified ways. In other words, the Tukey-Huber contamination model posits a representation of the data as a mixture distribution with a fully described dominant component and an unspecified minority component. Under this framework, the tasks of fitting a statistical model to a given dataset robustly (i.e. finding a fit close to the one we would have had in the absence of outliers) and that of anomaly detection (identifying outlying observations) are essentially equivalent problems [Hubert et al., 2008]. In the case of models designed to deal with potentially contaminated multivariate data, the Tukey-Huber contamination model will be assumed to apply to the full data-sets whereas in the case of models designed to deal with potentially contaminated functional data, the Tukey-Huber contamination model will be assumed to apply to the cross-sections (or multivariate sub-structures of the data).

In this thesis, I will often characterize the properties of statistical models and concepts in terms of their geometrical features. For example, to measure the robustness of an estimator to the presence of outliers in the data, we will often use the notion of finite sample breakdown point of an estimator, as introduced by [Donoho, 1982]. Given a sample and an estimator, this is the smallest proportion of observations that needs to be replaced by outliers to cause the estimated fit to be arbitrarily far from the values it would have had on the original sample. Remarkably, the finite sample breakdown point of an estimator can be derived without recourse to concepts of chance or randomness using geometrical features of a sample and the estimator alone. Another example of the geometric approach to statistics is the insistence on the characterization of estimators in terms of their equivariance with specific group of transformations of the data. Beside giving an intuitive meaning (even in higher dimensions) the quantity being estimated, equivariance properties are important for other reasons. The principle of equivariance (or Diffeomorphism) is an fundamental postulate in science [O'Hanian and Ruffini, 1980, pp 251–267]. Statistics is part of this intellectual traditions and, more pragmatically, so are many of the reasons data is collected and analyzed in the first place. Consequently, equivariance, or at the very least independence from the orientation of the coordinate system (a weaker requirement that makes sense in some high dimensional settings), is an important aspect in many statistical problems. Equivariance also plays a germane role in other aspects of the design of algorithms for robust estimation. For example, without equivariance, the concept of breakdown of an estimator (at least, as it is usually defined) needs to be interpreted with caution [Davies and Gather, 2005]. Likewise, the interpretation, and consequently the use of, simulations to measure the robustness of an estimators quantitatively is also to a great extent contingent upon its equivariance properties.

This thesis is also very focused on issues pertaining to the design and implementation of algorithms for (robust) data analysis. Particularly when imposing requirements in terms of equivariance and in the context of multivariate data the problem of fitting statistical models robustly is computationally hard [Bernholt, 2006]. Consequently, in most such applications, practitioners will often use approximations to the exact robust multivariate estimator and a large part of the research effort in robust statistics is devoted to the problem of designing and testing such approximation algorithms. In any case, perhaps to a even greater extent than in other branches of statistics [Rocke, 1998], software implementations play an important role in robust statistics. For this reason, an important part of the work involved in this thesis was spent on the production of testable, documented, portable and high performance implementations integrated in commonly used and open source statistical packages. The next three sections briefly describe the state of the art in the specific questions touched upon in the present thesis. Section 1.5 gives an outline of the individual chapters and formulates the main research aims of this thesis.

1.2 Robust Statistics in the context of multivariate data

Classical statistical methods are based on the assumption that a particular probability model generates the observed data. In the context of multivariate data the most commonly used assumption is that the observations are (multivariate) i.i.d. (identically and independent distributed) drawn from a symmetrical distribution. In the context of regression models the most commonly used assumption is that the data contains a causal (conditional) relationship linking a set of design variables to a second set of response variables up to i.i.d. residuals which are drawn from a common symmetrical distribution. It has long been recognized that either one of these assumptions are fairly restrictive and that one way to relax them is by substituting the assumption of identically distributed samples by the assumption that the data is a realization of a mixture distribution with a fully described dominant component and an unspecified minority component which we call outliers.

To fit such models in the context of estimating multivariate location vector and scatter matrix (in the context of multivariate problem) or regression parameters (in the context of multi-variable regression), several robust estimators were developed, starting in the eighties. In essence, these are methods that can find a fit for the majority component of the mixture without being unduly influenced by its minority component.

Considering only those procedures that have positive breakdown point and that can be run when p (the number of variables) is relatively large, the available robust procedures can be classified into two broad families. The first group consists of those algorithms that use a large number of random initial subsets. This guarantees exact affine equivariance of the procedure (i.e. the method behaves appropriately when the data are transformed linearly), but at the cost of a computation time that scales exponentially with the number of variables in the dataset.

In the context of estimators of location and scatter, the most important member of this group are the Stahel-Donoho estimator SDE (see e.g. [Maronna and Yohai, 1995]), the MCD and MVE estimators [Rousseeuw, 1984] and in particular their fast implementations FastMCD [Rousseeuw and Van Driessen, 1999] and FastMVE [Maronna et al., 2006, pp 199]. This group also contains several variants that use smooth (but non convex) loss functions such as the FastS algorithm [Salibian-Barrera and Yohai, 2006] and the FastMM algorithm [Salibian-Barrera and Yohai, 2006].

The second group is composed of estimators that shun the requirement of affine equivariance in order to obtain lower computation times. This group includes the BACON method [Billor et al., 2000], the orthogonalized Gnanadesikan-Kettenring (OGK) estimator [Maronna and Zamar, 2002], the DetMCD algorithm [Hubert et al., 2012] and the DetS and DetMM algorithms [Hubert et al., 2015c].

In the context of estimators of multi-variable regression, the most important member of this group are the LTS [Rousseeuw, 1984] as well as several alternatives that use smooth (but non convex) loss functions such as the regression S estimator [Rousseeuw and Yohai, 1984] and MM-estimator [Yohai, 1987] and in particular their popular "Fast" implementations, the FastLTS algorithm [Rousseeuw and Van Driessen, 2006] as well as the FastS and FastMM algorithms for regression [Salibian-Barrera and Yohai, 2006].

1.3 Robust Statistics in the context of functional data

In functional data analysis (FDA) one is usually interested in the analysis of a set of curves. To be more precise, typical measurements consist of N curves of the form $\{(t, Y_n(t))\}_{n=1}^{N}$ observed on an interval U. We assume that measurements are available in a discrete set of time points $t_1, t_2, \ldots, t_T$. Basic questions of interest are (i) the estimation of the central tendency of the curves,

(ii) the estimation of the variability among the curves, (iii) the detection of outlying curves, as well as (iv) classification or clustering of such curves.

A depth function provides an ordering from the center outwards such that the most central object gets the highest depth value and the least central objects the smallest depth. Some well-known depth functions are halfspace depth, simplicial depth, projection depth, zonoid depth, among others [Liu et al., 2006]. Recently, many notions of depth have been proposed for functional data, such as the Fraiman and Muniz (FM) depth [Fraiman and Muniz, 2001], random projection depth (RP) [Cuevas et al., 2007], band depth (BD) and modified band depth (MBD) [López-Pintado and Romo, 2009], and half-region depth [Lopez-Pintado and Romo, 2011]. All these depth functions are computed on the original set of observed curves $\{(t, Y_n(t))\}_{n=1}^N$. The FM depth and MBD depth are quite similar, as they both consider a *univariate* depth function at each time point t and define the functional depth as the *average* of these depth values over all time points.

To better handle shape differences, [Cuevas et al., 2007] have proposed to consider the original set of curves $\{(t, Y_n(t))\}_{n=1}^N$ as well as their derivatives $\{(t, Y_n'(t))\}_{n=1}^N$. They consider a number of random projections, project both sets of curves on each direction, apply a *multivariate* depth function on the bivariate sample and finally *average* the depth values over the random projections. Adding this extra information through the use of the derivatives is frequently done in FDA, see also [Ramsay and Silverman, 2005, Ferraty and Vieu, 2006].

To illustrate an example of bivariate functional data, we can look at a industrial data set from a process that produces one part during each cycle [De Ketelaere et al., 2011]. The behavior of the cycle as monitored by an accelerometer provides a fingerprint of the cycle and, related, of the quality of the produced part. If a deviating acceleration signal occurs, the process owner should be warned. The left plot in Figure 1.2 shows the acceleration signal of $N = 224$ parts measured during 120ms (in gray). Measurements are available every millisecond, hence the time signal ranges from $t_1 = 1$ up to $t_T = 120$. On this plot we see several curves with a deviating pattern, most prominently at the final stage of the production. The right plot in Figure 1.2 shows the corresponding velocity curves. Denoting the acceleration at time t_j as $A(t_j)$, the velocity at time t_j, is defined as $V(t_j) = \int_{-\infty}^{t_j} A(t)dt$ with $A(t)$ the acceleration at time t, we approximated the velocity by $V(t_j) \approx V(t_{j-1}) + (A(t_{j-1}) + A(t_j))/2$ starting with $V(t_1) = 0$.

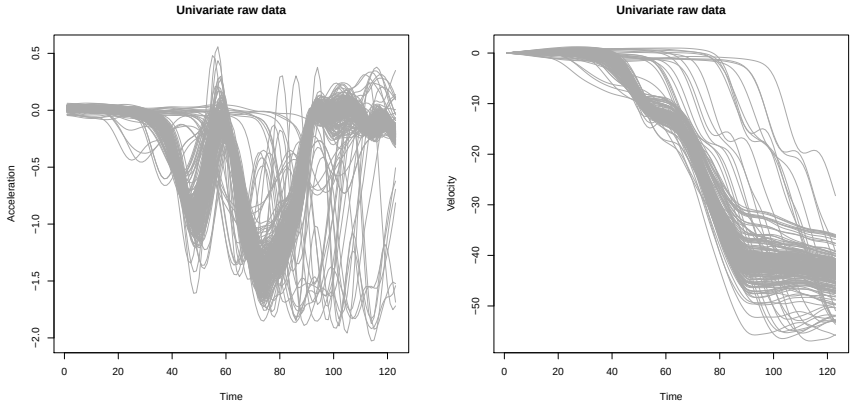


Figure 1.2: Industrial data: Acceleration (left) and Velocity (right) measured each millisecond, from $t_1 = 1ms$ up to $t_T = 120ms$.

1.4 Software packages for multivariate depths

Both the open source statistical software R [R Core Team, 2014] and the closed source matrix numerical computing environment Matlab [MATLAB, 2014] include documented open source libraries containing implementations of some important algorithms for computing depth based tools for estimation and visualizations of multivariate data. Let us note in particular the implementations in Libra [Verboven and Hubert, 2010] (on the Matlab side) and rdepth [Genest et al., 2012] and aplpack [Wolf and Bielefeld, 2014] on the R side. However, these libraries only included a subset of the most important multivariate depth concepts. For example, none of them included code to compute regression depth [Rousseeuw and Hubert, 1999] or depth based tests [Rousseeuw and Struyf, 2002] and [Van Aelst et al., 2002a]. Furthermore, the functions that were implemented in both libraries were not always implemented in a way to ensure that the results would always be consistent. In some cases (Outlyingness and Adjusted Oultyingness) modern implementations were available in both Matlab and R but these implementations could be rewritten to make the existing code more versatile, portable and nimble in terms of computational footprint, often by using state of the art high performance open source numerical libraries [Guennebaud et al., 2013]. Integrating many algorithms for depth and depth based visualizations in a single object oriented library also enables easier and more streamlined comparison of the result of the various algorithms, a key component of the so-called exploratory approach to data analysis [Tukey, 1962].

1.5 Outline of the Thesis

In Chapter 2, we compare several state of the art algorithms for multivariate outlier detection and estimation of location and scatter. Detecting outliers in a multivariate point cloud is not trivial, especially when dealing with a sizeable fraction of contamination. Over time, it has increasingly been recognized that the safest and most feasible approach to exposing outliers starts by computing a highly robust estimator of location and scatter that can withstand a large proportion of contamination. Many such estimators have been proposed in recent years. In Chapter 2, we will compare the worst-case bias of several prominent robust and affine equivariant multivariate estimators by means of a large simulation study. A related problem is also to compare the performance of two or more robust estimators on a real data set where the potential outliers are not identified. In this chapter, we present a new procedure to do this and use it to evaluate the performance of several state of the art robust estimator of location and scatter on four datasets.

Chapter 3 is devoted to a new algorithm designed to fit the linear regression model robustly, quickly and deterministically in context involving a large number of continuous design variables and a unique (also continuous) response variable. After presenting the proposed algorithm and its main properties (e.g. its equivariance and finite sample breakdown point) and comparing it to a robust but non deterministic alternative on a real data example, we discuss some possible alternative design choices for the proposed algorithm and justify of the chosen design. Chapter 4 is supported by a portable, testable, documented and fast implementation of the proposed algorithm integrated in the form of an east to use, install and distribute R package. This package contains the codes and dataset necessary to reproduce all the results shown in Chapter 4 as well as a set of easy to run and transparent R codes designed to test the correctness of the main components of the faster (but less accessible) C++implementation of the algorithm (also included in the package).

Chapter 4 deals with (multivariate) functional data and means to perform depth-based exploratory analysis on them. More precisely, a multivariate depth for functional data is defined and studied. By the multivariate nature and by including a weight function, it acknowledges important characteristics of functional data, namely differences in the amount of local amplitude, shape and phase variation. The multivariate sample of curves may include warping functions, derivatives and integrals of the original curves for a better overall representation of the functional data via the depth. A simulation study and data example confirm the good performance of this depth function.

Chapter 5 is devoted to an extended discussion of the mrfDepth package, a

joint R [R Core Team, 2014] and Matlab [MATLAB, 2014] software package. The mrfDepth package gathers many existing, as well as some improved, implementations of algorithms for carrying out depth based exploration analysis and inference on multivariate, functional and regression data sets. It improves on existing packages by gathering these functions in one place and using a unified framework to make available many algorithms that were otherwise put in different packages or not available in an easy to use, modern computing environment at all. We also improve on many of these implementations individually in terms of computational efficiency of the implementations as well as jointly, by integrating them together in an easy to expand object-oriented programming framework complete with examples and unified documentation.

Finally, Chapter 6 offer some closing discussion of the results obtained in this thesis.

Chapter 2

Shape Bias of Robust Covariance Estimators: An Empirical Study

2.1 Introduction

Given a collection of n column vectors $\boldsymbol{x}_i$ in $\mathbb{R}^p$, with $n > p$, the simplest and most general problem in multivariate analysis is that of estimating a location vector $\boldsymbol{\mu} \in \mathbb{R}^p$ and a scatter matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{p \times p}$. Many statistical methods rely on the gaussian maximum likelihood estimates $(\boldsymbol{t}_M, \boldsymbol{S}_M)$ of $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. These estimates are of course optimal when the $\boldsymbol{x}_i$ are drawn from a multivariate normal distribution, but suffer from their extreme sensitivity to outliers. To remedy this, several high breakdown estimators (i.e., methods that can withstand a large fraction of outliers without breaking down) were developed, starting in the eighties. We will compare the most commonly used methods by measuring how much the shape of (the confidence ellipsoids of) the scatter estimate can be biased by a given fraction of outliers.

In the following section we provide a brief introduction to the terminology and estimators. Section 2.3 outlines the scope and methodology of this chapter, while sections 2.4 and 2.5 present our main findings. Readers looking for a background on the basic concepts of multivariate robust estimation are referred to the books of [Rousseeuw and Leroy, 1987] and [Maronna et al., 2006].

2.2 Background

The available robust procedures can be classified into two broad families. The first group consists of the algorithms that use a large number of random initial subsets. This guarantees exact affine equivariance of the procedure (i.e. the method behaves appropriately when the data are transformed linearly), but at the cost of a computation time that scales as $O(2^p)$, prohibiting their use in dimensions larger than about $p = 12$. This group includes the Stahel-Donoho estimator SDE (see e.g. [Maronna and Yohai, 1995]), the MCD and MVE estimators [Rousseeuw, 1984] and in particular their fast implementations FastMCD [Rousseeuw and Van Driessen, 1999] and FastMVE [Maronna et al., 2006, pp 199]. It also contains variants using smooth (but non convex) loss functions such as the FastS algorithm [Salibian-Barrera and Yohai, 2006] and the FastMM algorithm [Salibian-Barrera et al., 2006]. The most often used MM methods are MM85 and MM95, where the latter attains higher efficiency for gaussian data. We are also including a two-step method which first runs FastMVE and then uses it as the single starting point for the FastS method. This method will be denoted as MVE_S throughout.

The second group is composed of estimators that shun the requirement of exact affine equivariance in favor of approximate equivariance, in order to obtain a much lower computation time. This group features the BACON method [Billor et al., 2000], the orthogonalized Gnanadesikan-Kettenring (OGK) estimator [Maronna and Zamar, 2002], and the DetMCD algorithm [Hubert et al., 2012].

We now briefly explain how these last two estimators work. We will denote the columns of $\boldsymbol{X}$ as $X_j,\ j = 1, \ldots, p$ and rows $\boldsymbol{x}_i,\ i = 1, \ldots, n$. The orthogonalized Gnanadesikan and Kettenring (OGK) estimates, [Huber, 1981, 202–204], [Maronna and Zamar, 2002] is a method to obtain a robust and positive definite scatter matrix from a matrix of robust pairwise correlation. When the procedure uses as starts the robust scatter estimate of [Gnanadesikan and Kettenring, 1972], the resulting multivariate location and scatter estimates are called orthogonalized Gnanadesikan and Kettenring (OGK) estimates and are calculated as follows:

1. Let $m(\cdot)$ and $s(\cdot)$ be robust univariate estimates of location and scale.
2. Construct $\boldsymbol{v}_i = \boldsymbol{D}^{-1}\boldsymbol{x}_i$, for $i = 1, \ldots, n$ with $\boldsymbol{D} = \mathrm{diag}(s(X_1), \ldots, s(X_p))$.
3. Compute the correlation matrix $\boldsymbol{U}$ of the columns of $\boldsymbol{V} = (V_1, \ldots, V_p)$ given by
$$u_{jk} = 1/4 \left(s^2(V_j + V_k) - s^2(V_j - V_k) \right). \tag{2.1}$$
4. Compute the matrix $\boldsymbol{E}$ of eigenvectors of $\boldsymbol{U}$ and

(a) project the data on these eigenvectors, i.e. $\boldsymbol{T} = \boldsymbol{VE}$;

(b) compute 'robust variances' of $\boldsymbol{T} = (T_1, \ldots, T_p)$, i.e. $\boldsymbol{\Lambda} = \text{diag}(s^2(T_1), \ldots, s^2(T_p))$;

(c) set $\boldsymbol{t}^0 = \boldsymbol{Em}$ where $\boldsymbol{m} = (m(T_1), \ldots, m(T_p))$ and compute the positive definite matrix $\boldsymbol{S}_0 = \boldsymbol{E\Lambda E}^\top$.

5. Transform back to the original scale of $\boldsymbol{X}$, i.e. $\boldsymbol{t}^{\text{OGK}} = \boldsymbol{Dt}^0$ and $\boldsymbol{S}^{\text{OGK}} = \boldsymbol{DS}^0\boldsymbol{D}^\top$.

Step 2 of the algorithm makes the estimates scale equivariant (by rescaling all the variables), whereas the next steps are a kind principal components that replace the eigenvalues of $\boldsymbol{U}$ (which may be negative definite) by robust variances.

The DetMCD estimates of location and scatter [Hubert et al., 2012] is obtained as follows. Denote $(\boldsymbol{t}^1, \boldsymbol{S}^1) = (\boldsymbol{t}^{\text{OGK}}, \boldsymbol{S}^{\text{OGK}})$. Given $(\boldsymbol{t}^1, \boldsymbol{S}^1)$ and 5 other deterministic candidate fits $(\boldsymbol{t}^j, \boldsymbol{S}^j)|j = 2, \ldots, 6$, the DetMCD estimates of location and scatter correspond to the candidate fit having smallest value of $|\boldsymbol{S}^j|$.

In many cases we give each observation a weight depending on its statistical distance $d(\boldsymbol{x}_i, \boldsymbol{t}, \boldsymbol{S}) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{t})^\top \boldsymbol{S}^{-1}(\boldsymbol{x}_i - \boldsymbol{t})}$ from the initial estimates $(\boldsymbol{t}, \boldsymbol{S})$. The final estimate is then the weighted mean and weighted covariance matrix of the observations, which tends to be more accurate at uncontaminated data. In order to compare apples to apples, we will use the reweighted versions of FastMCD, FastMVE, OGK and DetMCD, each with the cutoff set to its default value (these are 0.975 for all these estimators, except for OGK which uses 0.9). We set any other estimation parameter to its default value as in the `R` [R Core Team, 2014] package `rrcov` [Todorov and Filzmoser, 2009] and (for BACON) in `robustX` [Stahel and Maechler, 2009].

2.3 Methodology

2.3.1 Shape bias

We will generate many data sets as follows. The uncontaminated part of the data, denoted as $\boldsymbol{X}_u$, consists of $n - \lfloor \varepsilon n \rfloor$ observations generated from a normal distribution. The second part, denoted as $\boldsymbol{X}_c$, contains the remaining $\lfloor \varepsilon n \rfloor$ observations which are generated as outliers (in ways to be specified). The union of both parts is the contaminated data set denoted as $\boldsymbol{X}_\varepsilon$. I^C retains the index of the outliers in $\boldsymbol{X}_\varepsilon$.

For each contaminated data set $\boldsymbol{X}_\varepsilon$ we will measure how much its estimates $(\boldsymbol{t}, \boldsymbol{S})$ deviate from the true $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. For this we will focus on the shape component. The *shape matrix* of $\boldsymbol{\Sigma}$ is defined as $\boldsymbol{\Gamma} = |\boldsymbol{\Sigma}|^{-1/p}\boldsymbol{\Sigma}$. It follows that always $|\boldsymbol{\Gamma}| = 1$, and we can decompose the original matrix as $\boldsymbol{\Sigma} = |\boldsymbol{\Sigma}|^{1/p}\boldsymbol{\Gamma}$. The square root of this scalar factor, $|\boldsymbol{\Sigma}|^{1/2p}$, is called the scale component of $\boldsymbol{\Sigma}$. The shape matrix of the estimated scatter matrix S is computed analogously as $\boldsymbol{G} = |\boldsymbol{S}|^{-1/p}\boldsymbol{S}$, and its scale component is $|\boldsymbol{S}|^{1/2p}$. Many studies of bias have focused on the bias of the location estimate $\boldsymbol{t}$ or the bias of the scale component. Here we focus on the shape bias [Maronna and Yohai, 1990], which is defined as

$$\text{bias}(\boldsymbol{S}) = \log \frac{\lambda_1(\boldsymbol{G}^{-1/2}\boldsymbol{\Gamma}\boldsymbol{G}^{-1/2})}{\lambda_p(\boldsymbol{G}^{-1/2}\boldsymbol{\Gamma}\boldsymbol{G}^{-1/2})} \tag{2.2}$$

where $\lambda_1 \geq, \ldots, \geq \lambda_p$ are the eigenvalues. Obtaining a robust shape matrix $\boldsymbol{G}$ is the most important part of the robust estimation problem, since robust estimation of the scale component of $\boldsymbol{S}$ then becomes a simple univariate scale problem. Also, we can sphere the data with $\boldsymbol{G}^{-1/2}$ and estimate $\boldsymbol{t}$ by some simple location estimator such as the coordinatewise median.

2.3.2 Affine equivariant estimators

For affine equivariant estimators (AEE's) the shape bias depends on the dimension p and the contamination rate ε. It also depends on the 'distance' between the outliers and $\boldsymbol{X}_u$, which we will measure by

$$\nu = \min_{i \in I^C} d(\boldsymbol{x}_i, \boldsymbol{t}_u, \boldsymbol{S}_u) / \sqrt{\chi^2_{0.99,p}} \ . \tag{2.3}$$

It also depends on the spatial configuration of X_c. Given some constraints, we can get an idea about the worst adversary configuration. In increasing order of difficulty these are, for $X_u \sim N_p(\boldsymbol{\mu}_u, \boldsymbol{\Sigma}_u)$ and $X_c \sim N_p(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$:

- Shift configuration. If we are using the classical mean and covariance estimators and constrain the adversary to (a) $|\boldsymbol{\Sigma}_c| \geq |\boldsymbol{\Sigma}_u|$ and (b) place $\boldsymbol{X}_c$ at a distance ν of $\boldsymbol{X}_u$, then the adversary will set $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}_u$ (see Theorem 1 in [Rocke and Woodruff, 1996]) and set $\boldsymbol{\mu}_c$ in order to satisfy (b). Intuitively, this makes the components of the mixture the least distinguishable from one another.
- Point contamination. If we omit the constraint (a) above but keep (b), the adversary will place $\boldsymbol{X}_c$ in a single point so $|\boldsymbol{\Sigma}_c| = 0$ (see Theorem 2 in [Rocke and Woodruff, 1996]).

- If we omit both constraints (a) and (b), the adversary may set $\boldsymbol{\mu}_c = \boldsymbol{\mu}_u$ and choose $\boldsymbol{\Sigma}_c$ to obtain a large shape bias. The barrow wheel contamination [Stahel and Maechler, 2009] does this.

In the course of our study we also considered radial outliers and clustered outliers, but these situations were much easier to deal with so we do not show them.

2.3.3 Non affine equivariant estimators

For non affine equivariant estimators (NAEE's) the shape bias is not only affected by p, ε, ν, and the spatial distribution of the outliers, but also by the choice of $\boldsymbol{\Sigma}_u$. We have to resort to heuristic arguments to characterize difficult configurations. We now:

- replace the generic $\boldsymbol{\Sigma}_u$ by a matrix $\boldsymbol{DAD}^\top$ where $\boldsymbol{A}$ is a matrix with diagonal entries 1 and off-diagonal entries 0.75, and $\boldsymbol{D}$ is a diagonal matrix with diagonal entries drawn from the uniform distribution on $(0, 1)$.
- for point and shift-type contamination, we shift $\boldsymbol{\mu}_c$ along the eigenvector direction of $\boldsymbol{\Sigma}_u$ with smallest eigenvalue.

The three types of configuration are depicted in Figure 2.1 for $n = 100$, $p = 2$, $\varepsilon = 0.4$, and $\nu = 2$. The outlying observations are depicted as triangles. The blue cross and ellipse depict the MM95 estimates of location and scatter obtained from the `rrcov` package using the default settings.

2.3.4 Simulation parameters

We can generate the uncontaminated data $\boldsymbol{X}_u$ from a zero-mean distribution since all methods under consideration are location equivariant. For the shift and point configurations, the outliers are generated as $\boldsymbol{X}_c \sim N_p(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ where $\boldsymbol{\Sigma}_c$ is either $\boldsymbol{I}_p$ or $10^{-4} I_p$ (depending on whether the outliers are Shift or Point-Mass, respectively) and $\boldsymbol{\mu}_c$ is a scalar multiple of the last eigenvector of $\boldsymbol{S}_u$. We then measure ν in (2.3). The barrow wheel configuration is generated by the `robustX` package [Stahel and Maechler, 2009] with default parameters. Here is the complete list of simulation parameters:

- the dimension p is one of $\{4, 8, 12, 20\}$,
- the sample size is $n = 25p$,

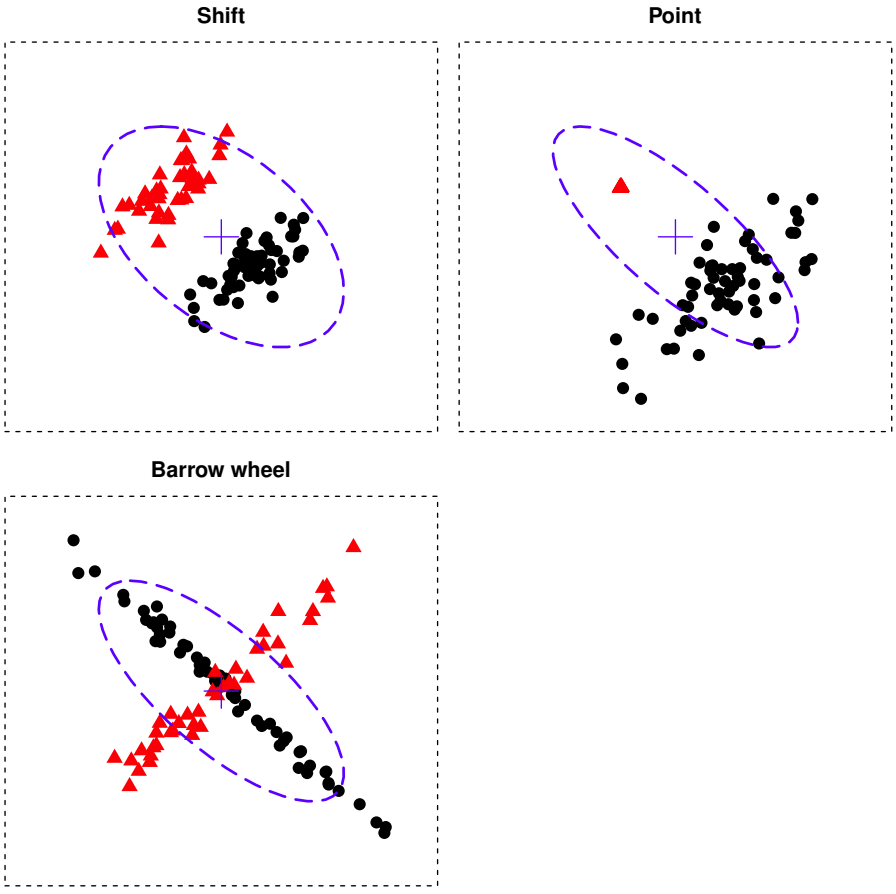


Figure 2.1: The three outlier configurations, together with the MM-95 estimates of location (cross) and scatter (ellipse). The outliers are depicted as triangles.

- the contamination fraction ε is one of $\{0.1, 0.2, 0.3, 0.4\}$,
- the configuration of the outliers is either shift, point, or barrow wheel,
- for shift and point contamination, the distance ν comes from the uniform distribution on (0,10). The barrow wheel contamination does not depend on ν.
- the number of initial $(p+1)$-subsets N_s (for the AEE's) is given by:

$$N_s = \frac{\log(0.01)}{\log(1-(1-\varepsilon)^{p+1})} \tag{2.4}$$

  when p is one of $\{4, 8, 12\}$ and $\varepsilon = 0.4$. This says that the probability of getting at least one uncontaminated initial subset is at least 99%. When $p = 20$, the number given by application of Equation (2.4) becomes too large so N_s is capped at $N_s = 5000$.

We will display the results graphically in Figures 2.2 to 2.4. The response variable will be the shape bias. The variables p and the contamination type are discrete, so each panel has one combination. The bias increased monotonically with ε, so that not much information is lost by showing graphs for just a few values of ε (0.2 and 0.4). For barrow wheel contamination (Figure 2.5), these are all the parameters we have. The shift and point contaminations also depend on ν. The behavior of the bias as a function of ν is more difficult to foresee. Clearly, detecting nearby outliers can be more difficult than far away ones (if the outliers are far enough, a simple coordinatewise approach should be able to find them). On the other hand, nearby outliers don't cause as large an effect on bias as far away ones. This explains our choice of ν as the sole variable allowed to vary continuously over a range, on the horizontal axis of each panel in Figures 2.2 and 2.3. Note that each panel in these figures is based on 1000 simulation runs.

2.4 Simulation results

2.4.1 Results for affine equivariant estimators

We first compare the empirical bias of the AEE's under our set of outlier configurations. When $\varepsilon = 0.1$ all of these methods fared equally well, so we omit those results here.

The first and second rows of the lattice plots in Figure 2.2 show the bias behavior of the AEE's for $\varepsilon = 0.2$ and various dimensions p, from $p = 4$ (leftmost column)

to $p = 20$ (rightmost column). Inside each panel, the curves show the bias as a function of ν. (More precisely, the interval $[0, 10]$ was divided in 20 equispaced bins, and in each the median bias was computed.) The first row is for shift contamination and the second for point contamination. The method labeled MSDE is a modification (to be described below) of the Stahel-Donoho estimator.

In the first row of Figure 2.2 we see that shift contamination already reveals some differences between the methods. Starting from $p = 8$, the bias of FastS, MM85 and MM95 has a "bump" around $\nu = 2$. From $p = 12$ onward, these three methods have significantly higher bias than SDE, FastMCD and FastMVE for most ν. Note that the bias curve of FastMCD is hidden by that of FastMVE, while the curves for FastS, MM95 and MM85 are on top of each other.

Point contamination has a bigger effect. In the second row of Figure 2.2 we see that for $p \geq 8$ FastS, MM85, MM95 and FastMCD get a large bias. From $p = 12$ onward, only MSDE, FastMVE and MVE_S have low bias. Point contamination has been given the most attention in the robust literature, and our results are qualitatively similar to Table 1 of [Maronna and Zamar, 2002]: FastMCD has a larger bias than MSDE and FastMVE, and the difference increases with p and ν.

Finally, the results for the barrow wheel contamination are depicted in the first row of Figure 2.5. Since this configuration is not a function of ν, the shape bias of each method is summarized by a skewness-adjusted boxplot [Hubert and Vandervieren, 2008]. For low dimensions p the estimators do better than under point contamination. But from $p = 12$ onward this configuration generates the largest bias for FastS, MM85, MM95, and (as p increases to 20) for FastMCD, while not much affecting MSDE and FastMVE.

Increasing ε to 0.4 as in Figure 2.3 yields qualitatively different results. We first consider dimensions $p \in \{4, 8, 12\}$, for which the number of subsets is sufficient to ensure that at least one of the initial subsets is clean. For shift contamination, the bias of all methods gets higher for some ν, except for FastMCD which remains reliable. For point contamination all AEE's exhibit large bias. Under barrow wheel contamination (shown in the second row of Figure 2.5 for this value of ε), all biases get high except for the MSDE at $p = 4$. In the $\varepsilon = 0.3$ case (not shown) MSDE and FastMVE are the only ones to remain reliable under the barrow wheel for all p.

When $p = 20$, Equation (2.4) yields an intractably large number of initial $(p + 1)$-subsets so we cap their number at 5000. This is too few to ensure at least one clean subset and, clearly, starting from contaminated subsets renders all AEE's unreliable, as we can see from their bias plots.

To further explore this question, we tested a 'cheating' version of FastMCD

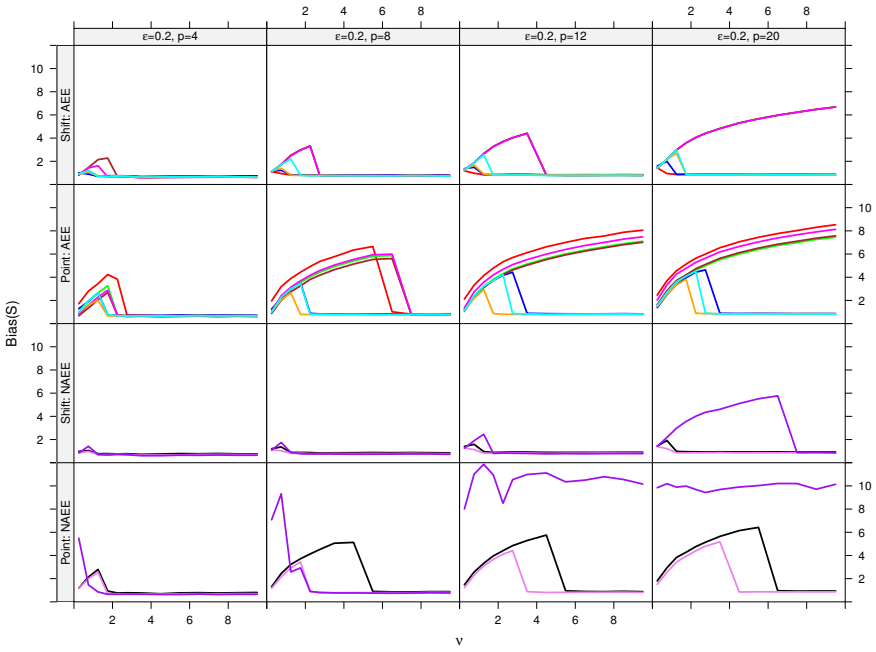


Figure 2.2: Empirical biases of our panel of estimators for $\varepsilon = 0.2$ against various types of contamination and values of p: FastMCD, FastMVE, FastS, MM95, MM85, MSDE,MVE_S,OGK, DetMCD,BACON.

where one of the 5000 initial subsets is replaced by a clean one (by drawing $p+1$ points from X_u). The bias of this cheating method is shown in Figure 2.4 for $p = 20$ and $\varepsilon \in \{0.2, 0.4\}$. Contrasting these results with the earlier ones we see that the absence of a clean initial subset is what caused the bias of FastMCD for shift contamination with high ε, but is not the whole explanation for point and barrow wheel contamination.

Finally, we note that the Stahel–Donoho estimator (or more specifically the weights used in it) had a problem for point and barrow wheel contamination when $\varepsilon = 0.4$. The estimated $\boldsymbol{S}$ was singular 25% of the time for point contamination with low p, up to 98% of the time for barrow wheel contamination with high p. To explain what happened, let us recall the method's definition. First, the outlyingness u_i of each observation $\boldsymbol{x}_i$ is computed by means of many projections. Next, a smooth weight function w is applied to these u_i. The final estimates are then the weighted mean and covariance matrix of the observations $\boldsymbol{x}_i$ with weights $w(u_i)$. However, for $\varepsilon = .4$ it often happened that the denominator

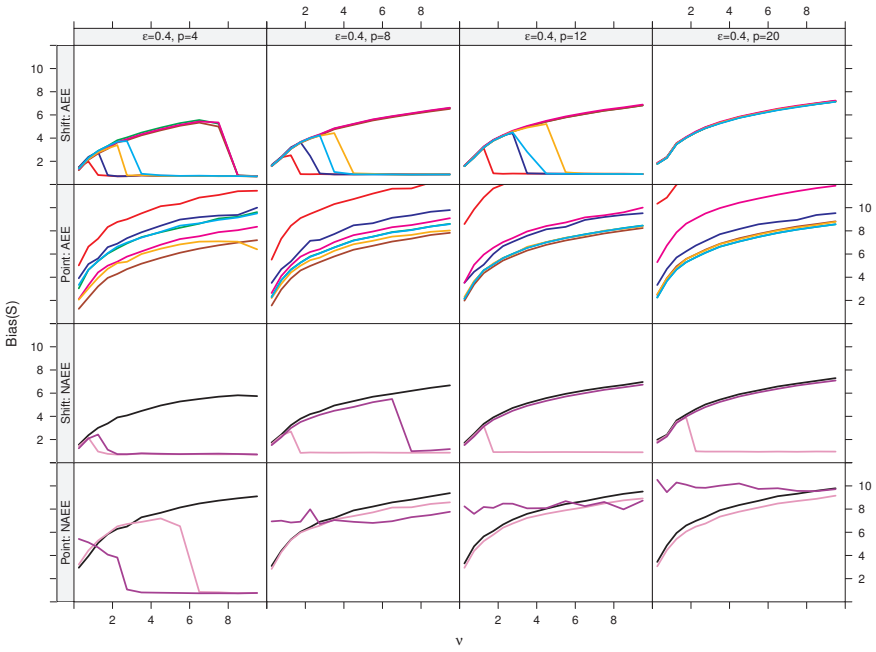


Figure 2.3: Empirical biases of our panel of estimators for $\varepsilon = 0.4$ against various types of contamination and values of p: FastMCD, FastMVE, FastS, MM95, MM85, MSDE, MVE_S, OGK, DetMCD, BACON.

of the weight was made arbitrarily large by the outliers, so that fewer than $p + 1$ observations were given a weight $w(u_i) > 0$ up to numerical precision, leading to a singular estimated $\boldsymbol{S}$. We remedied this problem by replacing this smooth function w by weights that are set to 1 for the h points with lowest outlyingness, and to 0 for the others. This goes back to [Hubert et al., 2005] and [Debruyne and Hubert, 2009] and ensures that enough data points are included for nonsingularity (assuming the uncontaminated data were in general position). The resulting MSDE method had a lower bias than the original Stahel-Donoho estimator throughout.

The main conclusion for AEE's seems to be that for a mild contamination fraction (say $\varepsilon \leqslant 0.2$) FastMVE is a good compromise method, typically yielding the second smallest bias (whereas the smallest bias method tends to differ depending on the unknown contamination type). The MVE_S method is quite similar to FastMVE. For higher ε AEE's generally have a large bias, with the notable exception of FastMCD in the case of shift contamination, when the

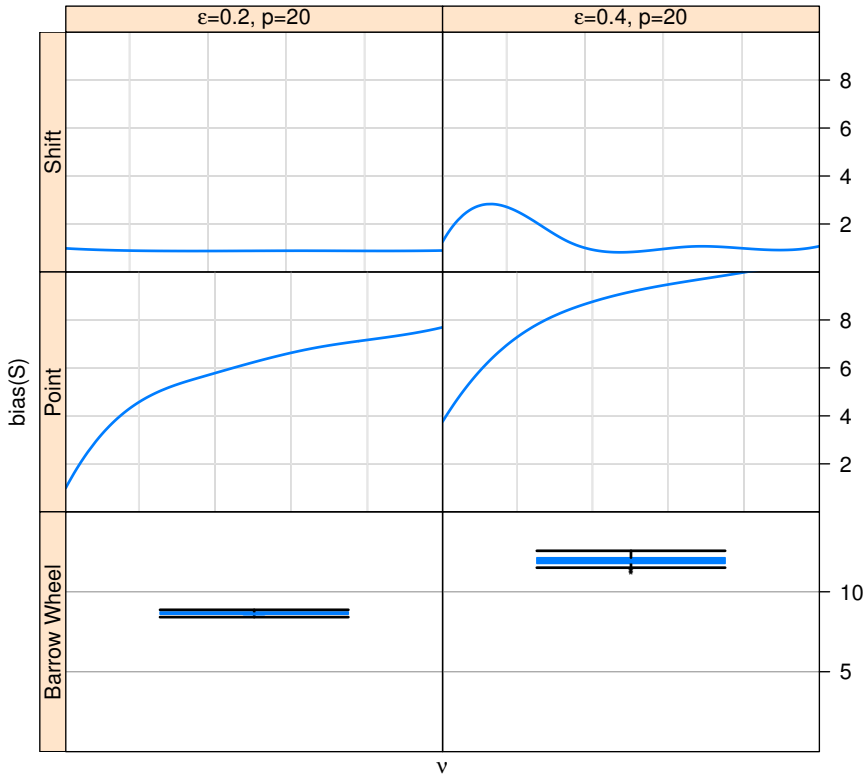


Figure 2.4: Empirical bias of the cheating FMCD estimator for our 3 configuration of outliers and $p = 20$ for $\varepsilon = 0.2$ and $\varepsilon = 0.4$

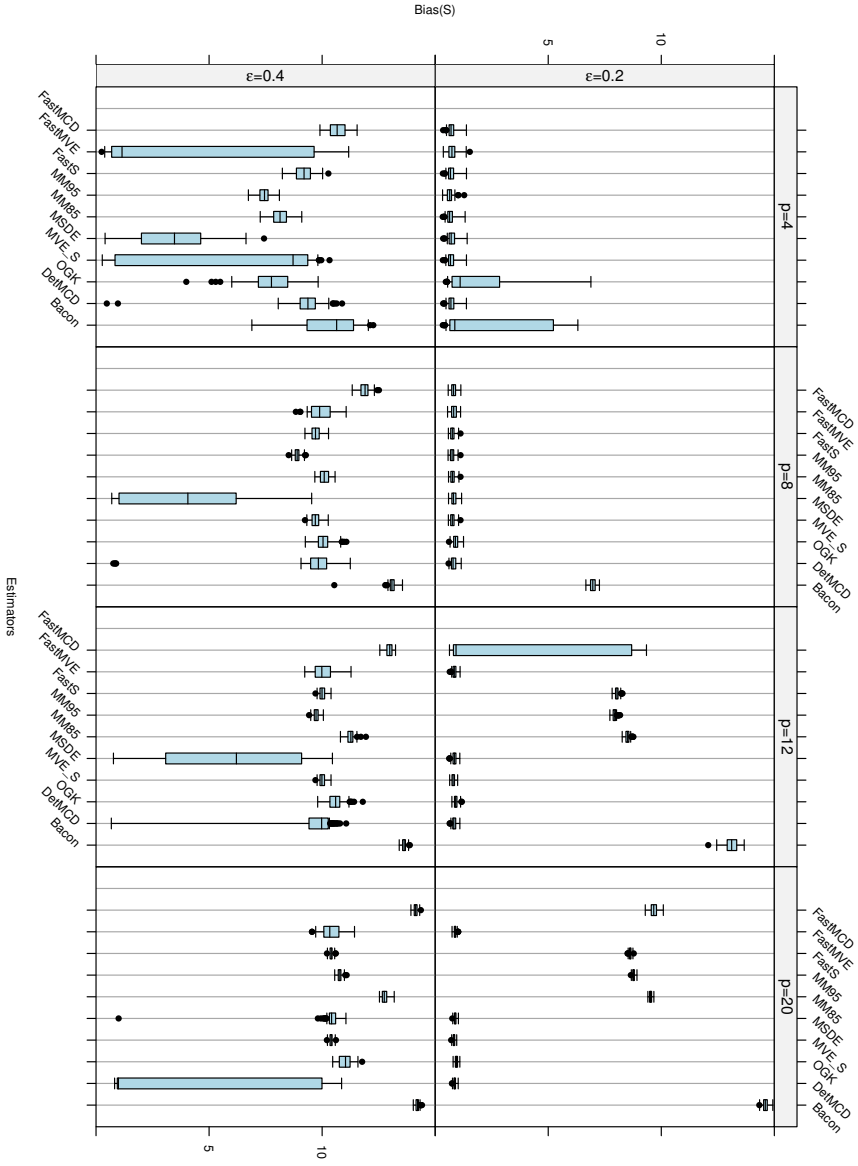


Figure 2.5: Empirical biases against the barrow wheel configuration for various contamination rates and values of p.

dimension is low enough to ensure at least one clean initial subset.

2.4.2 Results for non affine equivariant estimators

For the NAEE's we start with the case $\varepsilon = 0.2$ in rows two and four of Figure 2.2. For shift contamination all NAEE's do well, except for BACON when $p = 20$. For point contamination DetMCD performs fine, with bias comparable to that of FastMVE but not quite as low as SDE. The bias of OGK is systematically higher than that of DetMCD, and BACON does badly. Under barrow wheel contamination (first row of figure 2.5) DetMCD performs as well as MSDE and FastMVE, while BACON fails.

When we increase ε to 0.4 we see in the second row of Figure 2.3 that DetMCD is the only NAEE to remain reliable against shift contamination. All NAEE's do poorly against point contamination with $\varepsilon = 0.4$ and barrow wheel contamination (second row of Figure 2.5), but no worse than the AEE's.

Overall, we find that BACON does not perform well. The bias of OGK and DetMCD is similar to the best AEE's, with the exception of DetMCD at shift contamination where it outperforms all other methods. For high p and high ε it is infeasible for AEE's to run enough initial subsets to ensure at least one of them is clean, and in that case DetMCD and OGK are the only options.

2.5 Comparisons on real data

2.5.1 Results for affine equivariant estimators

Rather than looking at worst-case performance as in the simulation, we will compare methods on four real data sets. These are the Philips data [Rousseeuw and Van Driessen, 1999], the Pulp Fiber dataset [Rousseeuw et al., 2004], the Milk Composition Data [Daudin et al., 1988] and the Google Flu data. The latter contains the log of the number of influenza related search queries on Google in five countries (Austria, Belgium, Canada, France, and Germany) for 100 time periods covering 30/09/2003 to 01/03/2012 [Trends, 2012]. An earlier version of these data was analyzed by [Ginsberg et al., 2009] who reported that influenza search queries are strongly correlated with outbreaks of influenza.

To compare two methods on a real data set, let us denote their results as $(\boldsymbol{t}_j, \boldsymbol{S}_j)$ and $(\boldsymbol{t}_k, \boldsymbol{S}_k)$. Then denote H_j the set of $h = \lfloor (n + p + 1)/2 \rfloor$ observations with smallest values of $d^2(\boldsymbol{x}_i, \boldsymbol{t}_j, \boldsymbol{S}_j)$. Let $H_{jk} = H_j \cap H_k$. For all observations $\boldsymbol{x}_i$ in

H_{jk} we then compute the ratio of their likelihoods in models j and k:

$$R(i,j,k) = \log\left(\frac{L(\boldsymbol{x}_i, \boldsymbol{t}_k, \boldsymbol{S}_k)}{L(\boldsymbol{x}_i, \boldsymbol{t}_j, \boldsymbol{S}_j)}\right) = \log\left(\frac{d^2(\boldsymbol{x}_i, \boldsymbol{t}_j, \boldsymbol{S}_j))|\boldsymbol{S}_j|}{d^2(\boldsymbol{x}_i, \boldsymbol{t}_k, \boldsymbol{S}_k))|\boldsymbol{S}_k|}\right) \tag{2.5}$$

Typically, if both $(\boldsymbol{t}_j, \boldsymbol{S}_j)$ and $(\boldsymbol{t}_k, \boldsymbol{S}_k)$ are (un)affected by outliers, the members of H_{jk} are (un)contaminated so we would not expect them to have systematically higher likelihoods in either model. In other words, one would expect the $R(i,j,k)$ in Equation (2.5) to be fairly symmetrically distributed about 0. On the other hand, if one of the estimators is unaffected by the outliers and the other is affected, the members of H_{jk} are clean, and they will tend to have a higher likelihood under the unaffected estimates than under the affected ones. Therefore the $R(i,j,k)$ in Equation (2.5) will on average be less than 0 if $(\boldsymbol{t}_j, \boldsymbol{S}_j)$ is the unaffected method, and greater than 0 otherwise.

Figure 2.6 shows the skewness-adjusted boxplots for the $R(i,j,k)$ for all pairs of AEE's, for each of the four datasets. When the boxplot (and in particular its median) lies to the left of 0 it means the first-mentioned method did better.

As expected, the comparisons are less clear-cut than in the simulation. This is partly due to the fact that two of the datasets (Milk and Pulp Fiber) have a small n/p ratio (they have 86 and 62 data points, with $p = 8$ in both), and also because the outlier configurations were obviously not designed to be worst-case. But still the real data comparisons confirm the results of the simulation, at least qualitatively. There is strong evidence that MM95 does not do as well as the other estimators, including MM85 and FastS. This is consistent with [Salibian-Barrera et al., 2006] who report that in higher-dimensional settings MM95 no longer yields an improvement over the initial S, and even has an adverse effect. There is also weaker evidence that FastMCD behaves better than MSDE, FastMVE and MVE_S. This is because point contamination, on which MSDE excels, is not present in these data sets. Indeed, plotting projections of the data on the first three principal components of the robust correlation matrices obtained from MSDE and FastMCD indicates that the outliers are distributed among several small and well-separated clusters (Philips, Google Flu) or are of the shift variety (Milk, Pulp Fiber). These outlier configurations are more favorable to FastMCD in simulation.

2.5.2 Results for non affine equivariant estimators

The results of the real data comparisons in Figure 2.7 are again in line with the simulation, taking into account that these data sets do not contain point contamination. We see that DetMCD does slightly better than OGK, but not by much. Clearly, BACON is the worst of the three NAEE's.

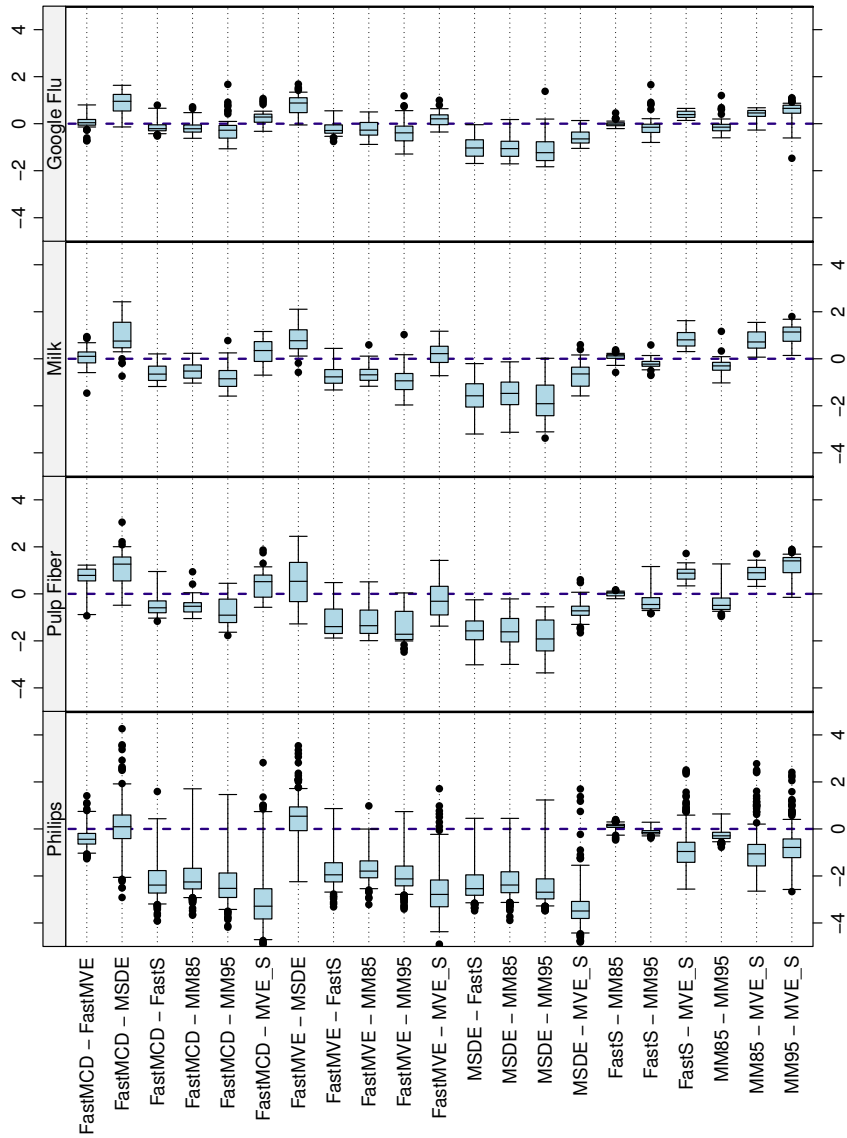


Figure 2.6: Adjusted boxplots of the $R(i, j, k)$ for various robust estimators and datasets.

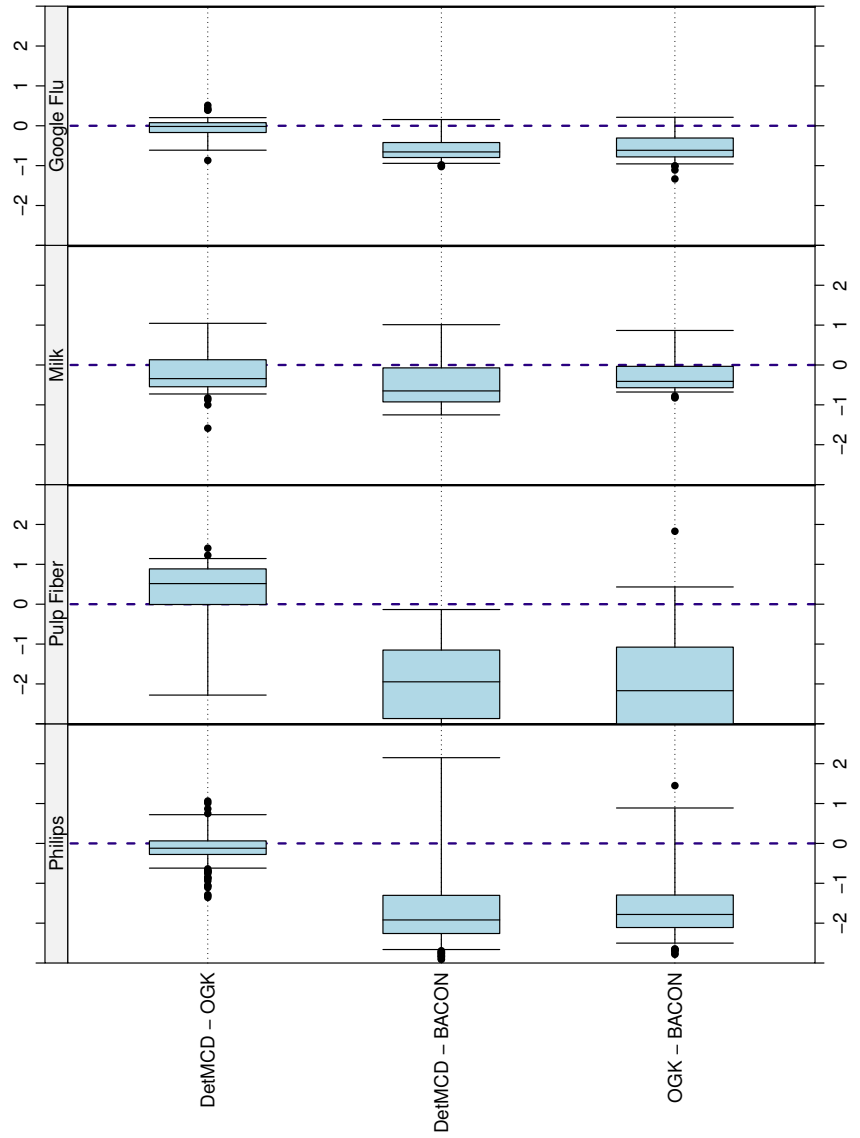


Figure 2.7: Adjusted boxplots of the $R(i, j, k)$ for various robust estimators and datasets.

2.6 Discussion

We have compared nine state-of-the-art affine and non affine equivariant estimators under various outlier configurations and for different dimensions and contamination rates.

When the contamination rate ε is up to 10%, all of these estimators had a low bias for all the dimensions and contamination types in our study. But for higher ε we started to see substantial differences in the shape bias. Already for $\varepsilon = 0.2$ we saw that FastS and FastMM had high bias in cases where FastMVE, MSDE and FastMCD did not. Also, point contamination had a substantially bigger effect on FastMCD than on MSDE and FastMVE, and this gap increases with p which is in line with the literature.

At our highest contamination rate ($\varepsilon = 0.4$), FastMCD was the only method capable of withstanding shift contamination (and it turns out that this is already true for $\varepsilon = 0.3$).

We also found that for all AEE's it is critical to get at least one clean initial subset. If ε and p are too high for this (at $p = 20$ this happens for $\varepsilon \geq 0.3$) all AEE's get a high bias.

Among non affine equivariant estimators (NAEE's), for $\varepsilon = 0.2$ both DetMCD and OGK have lower bias than BACON. For point and barrow wheel contamination they have lower bias than FastMCD, and are on par with MSDE. When $\varepsilon = 0.4$ all the NAEE's fail against point contamination (and this is already the case for $\varepsilon = 0.3$ at the larger dimensions p). Against high-ε shift and barrow wheel configurations DetMCD is the best NAEE, and it also outperforms FastMCD when $p = 20$ (since the latter does not get any clean initial subsets).

Unlike the worst-case contaminations in the simulation, the real data sets offer a more typical setting. It turned out that in the data sets under consideration the outliers were of the shift type or formed clusters. As expected, FastMCD was the best AEE in this case. Among the NAEE's, DetMCD did slightly better than OGK, which in turn greatly outperformed BACON.

In light of these results, our recommendation to practitioners is to choose the estimation method according to the dimension of their dataset. When $p \leqslant 10$ (or perhaps $p \leqslant 12$) our advice is to run FastMVE or FastMCD, and preferably both so they can be compared. When p is larger than this it becomes harder or even infeasible to draw enough initial subsets, and then we recommend to run DetMCD.

The deviations between $(\boldsymbol{t}, \boldsymbol{S})$ and $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be decomposed unto its shape, location and scale components. The measure of bias we used in this chapter takes

into account the former two components of the bias (through the dependence of $\boldsymbol{S}$ on $\boldsymbol{t}$) but ignores, by construction, the scale component of the fit. For all estimators, the scale component is typically estimated using a so-called re-weighting step which depends on $(\boldsymbol{t}, \boldsymbol{S})$ as well as the assumed distribution of the data. The different algorithms we compare use a variety of different re-weighting steps proposed over the years. For the purpose of the comparison carried in this Chapter we simply used, for each algorithm, it's own re-weighting step, assuming that the designer of each algorithm had already chosen the re-weighting step best attuned to it. Of course, it could be interesting to also compare these re-weighting steps, for example using a measure of scale bias such as $\log^2\left(\frac{|\boldsymbol{S}|}{|\boldsymbol{\Sigma}|}\right)$.

Chapter 3

Deterministic Algorithms for Robust Regression

3.1 Introduction

Consider the ordinary linear model:

$$y_i = \boldsymbol{x}_i^\top \boldsymbol{\beta} + \epsilon_i \tag{3.1}$$

where $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,p-1}, 1)^\top$ and $y_i \in \mathbb{R}$ and the first $p-1$ entries of $\boldsymbol{x}_i$ and y_i are both drawn from continuous distributions and $\epsilon_i \sim$ i.i.d. $\mathcal{N}(0, \sigma^2), \sigma^2 > 0$. Given a sample $(\boldsymbol{X}, Y) = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ with $n >> p$ it is well known that the usual least squares estimator (LS) of $\boldsymbol{\beta}$ possesses many desirable properties when all n observations come from model (3.1). However, in applications, we often encounter situations where the sample also contains a –typically unknown but small– proportion of data points that do not follow model (3.1). We call such non-conforming data outliers and it is also well known that the LS estimator is extremely sensitive to them. For example, a few outliers suffice to drive the LS estimates of $\boldsymbol{\beta}$ to any arbitrary value. In contrast, the least trimmed squares (LTS) method [Rousseeuw, 1984], the S [Rousseeuw and Yohai, 1984] and MM-estimator [Yohai, 1987] are three highly robust estimators of $\boldsymbol{\beta}$: they all attain a high finite sample breakdown point (i.e. $(n-p+1)/(2n) < 1/2$) meaning that these estimates of $\boldsymbol{\beta}$ always stay in a bounded region whenever $[(n-p)/2]$ or fewer observations are replaced by arbitrarily values.

However, in most cases, computing the exact LTS, S or MM fits turns out to be too computationally demanding and in applications practitioners will often use

the FastLTS [Rousseeuw and Van Driessen, 2006], the FastS [Salibian-Barrera and Yohai, 2006] or FastMM algorithms instead (henceforth the 'Fast' family of algorithms), which are stochastic approximations to the exact LTS, S and MM estimators respectively. Nonetheless, the computational cost of obtaining the FastLTS, FastS and FastMM fits, while much lower than that of their exact counterpart, still becomes prohibitive when p is large.

In this chapter we introduce DetLTS, DetS and DetMM (henceforth the 'Det' family of algorithms). In essence, as we explain in detail in Section 3.3, we propose to replace the multitude of random starting points used in the first step of the 'Fast' algorithms by a unique and deterministic one. As we show in Theorem 1 below, our approach will preserve the high breakdown point while being even faster to compute than either the FastLTS, FastS and FastMM algorithms. The use of a single deterministic start causes the new algorithms to lose some of the desirable properties of FastLTS, FastS and FastMM (for example, the new algorithms are not fully affine and regression equivariant) but also to gain some others in return. For example, like LS, the DetLTS, DetS and DetMM fits are deterministic (for a given dataset they will always yield the same solution) and permutation invariant (permuting the order of the observations in the data does not change the value of the DetLTS, DetS or DetMM fit). The approach we propose shares some features with that used in the context of robust estimation of scatter and location by the DetMCD [Hubert et al., 2012], DetS [Hubert et al., 2015c] algorithms where the multitude of random starting points used in the first step of the FastMCD [Rousseeuw and Van Driessen, 1999] and FastS [Todorov and Filzmoser, 2009] algorithms (used to approximate the MCD [Rousseeuw, 1984] and multivariate S [Rousseeuw and Leroy, 1987, p. 263] solution respectively) are replaced by six deterministic ones.

The new algorithms we propose combine ideas from many existing algorithms as building blocks and we first briefly recall these in Section 3.2. In Section 3.3 we detail the three new algorithms. Section 3.4 illustrates the use of the new algorithm on a real data application.

3.2 Algorithms for robust regression

3.2.1 General description of the new algorithms

We begin with a short outline of the main ideas behind the DetLTS, DetS and DetMM algorithms. To find a robust estimates for $\boldsymbol{\beta}$ and σ, the parameters in Model (3.1), we proceed as follows. First, we construct an initial estimate $\hat{\boldsymbol{\beta}}^{\text{init}}$ (the discussion of how $\hat{\boldsymbol{\beta}}^{\text{init}}$ is computed is deferred to the next section) and

consider the corresponding n-vector of residuals:

$$r_i^{\text{init}} = r_i(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{init}}) = y_i - \boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}^{\text{init}} \tag{3.2}$$

Next, we use a bounded function of these residuals as weights in a local improvement algorithm which we run until convergence. We consider two different local improvement procedures: the regression C-step (with intercept adjustment) and the regression I-step (both are also detailed in the subsequent subsections). The corresponding vector of fitted parameters we call the raw DetLTS or DetS fit depending on whether the local improvement step is the regression C-step or the I-step. Finally, we apply a re-weighting step (to the raw DetLTS fit) or an MM-step (to the DetS fit) to obtain $(\hat{\boldsymbol{\beta}}^{\text{DetLTS}}, \hat{\sigma}^{\text{DetLTS}})$ and $(\hat{\boldsymbol{\beta}}^{\text{DetMM}}, \hat{\sigma}^{\text{DetMM}})$ respectively the final (re-weighted) DetLTS and DetMM estimates of $\boldsymbol{\beta}$ and σ. In the next subsection, we recall briefly the basic building blocks of the algorithms we propose, deferring the actual discussion of the 'Det' algorithms themselves to Section 3.3. From now on, denote as $\boldsymbol{Z}$ the n by p data matrix formed of the first $p-1$ columns of $\boldsymbol{X}$ and the n-vector Y, h an integer satisfying $n \geqslant h \geqslant [\frac{n+p+1}{2}]$.

We will denote the columns of $\boldsymbol{Z}$ as $Z_j,\ j = 1, \ldots, p$ and rows $\boldsymbol{z}_i,\ i = 1, \ldots, n$. Likewise, for any n vector Y and integer $h \leqslant n$, $Y_{(h)}$ is the h-th order of statistics of the entries of Y.

We will also often discuss the various relevant concepts of equivariance in the context of regression estimators. A regression estimator $\hat{\boldsymbol{T}}(\boldsymbol{X}, Y)$ of $\boldsymbol{\beta}$ is affine equivariant if for any dataset $(\boldsymbol{X}, Y)$ it holds that:

$$\boldsymbol{\Pi}\hat{\boldsymbol{T}}(\boldsymbol{X}\boldsymbol{\Pi}, Y) \quad = \quad \hat{\boldsymbol{T}}(\boldsymbol{X}, Y) \tag{3.3}$$

where

$$\boldsymbol{\Pi} \quad = \quad \begin{pmatrix} \boldsymbol{Q} & \mathbf{0}_{p-1} \\ \mathbf{0}_{p-1}^\top & 1 \end{pmatrix}.$$

and $\boldsymbol{Q}$ is any non-singular $(p-1) \times (p-1)$ matrix. Likewise, a regression estimator $\hat{\boldsymbol{T}}(\boldsymbol{X}, Y)$ of $\boldsymbol{\beta}$ is regression equivariant if for any $n \times (p+1)$ dataset $(\boldsymbol{X}, Y)$, it holds that:

$$\hat{\boldsymbol{T}}(\boldsymbol{X}, Y + \boldsymbol{X}\boldsymbol{b}) - \boldsymbol{b} \quad = \quad \hat{\boldsymbol{T}}(\boldsymbol{X}, Y) \tag{3.4}$$

where $\boldsymbol{b}$ is any p-vector. A regression estimator $\hat{\boldsymbol{T}}(\boldsymbol{X}, Y)$ of $\boldsymbol{\beta}$ is permutation invariant if for any $n \times (p+1)$ dataset $(\boldsymbol{X}, Y)$, it holds that:

$$\hat{\boldsymbol{T}}(\boldsymbol{P}(\boldsymbol{X}, Y)) \quad = \quad \hat{\boldsymbol{T}}(\boldsymbol{X}, Y) \tag{3.5}$$

for any permutation matrix $\boldsymbol{P}$. A permutation matrix is a square matrix that has a single entry 1 in each row and each column, and zeros elsewhere. Therefore, $\boldsymbol{P}(\boldsymbol{X}, Y)$ simply permutes the rows of $(\boldsymbol{X}, Y)$.

3.2.2 The OGK estimator of multivariate scatter

The orthogonalized Gnanadesikan and Kettenring (OGK) estimates, [Huber, 1981, 202–204], [Maronna and Zamar, 2002] is a method to obtain a robust and positive definite scatter matrix from a matrix of robust pairwise correlation. When the procedure uses as starts the robust scatter estimate of [Gnanadesikan and Kettenring, 1972], the resulting multivariate location and scatter estimates are called orthogonalized Gnanadesikan and Kettenring (OGK) estimates and are calculated as follows:

1. Let $m(\cdot)$ and $s(\cdot)$ be robust univariate estimates of location and scale.
2. Construct $\boldsymbol{v}_i = \boldsymbol{D}^{-1}\boldsymbol{z}_i$, for $i = 1, \ldots, n$ with $\boldsymbol{D} = \mathrm{diag}(s(Z_1), \ldots, s(Z_p))$.
3. Compute the correlation matrix $\boldsymbol{U}$ of the columns of $\boldsymbol{V} = (V_1, \ldots, V_p)$ given by
$$u_{jk} = 1/4 \left(s^2(V_j + V_k) - s^2(V_j - V_k) \right). \tag{3.6}$$
4. Compute the matrix $\boldsymbol{E}$ of eigenvectors of $\boldsymbol{U}$ and
   (a) project the data on these eigenvectors, i.e. $\boldsymbol{T} = \boldsymbol{V}\boldsymbol{E}$;
   (b) compute 'robust variances' of $\boldsymbol{T} = (T_1, \ldots, T_p)$, i.e. $\boldsymbol{\Lambda} = \mathrm{diag}(s^2(T_1), \ldots, s^2(T_p))$;
   (c) set $\hat{\boldsymbol{\mu}}(\boldsymbol{V}) = \boldsymbol{E}\boldsymbol{m}$ where $\boldsymbol{m} = (m(T_1), \ldots, m(T_p))$ and compute the positive definite matrix $\hat{\boldsymbol{\Sigma}}(\boldsymbol{V}) = \boldsymbol{E}\boldsymbol{\Lambda}\boldsymbol{E}^\top$.
5. Transform back to the original scale of $\boldsymbol{Z}$, i.e. $\hat{\boldsymbol{\mu}}^{\mathrm{OGK}} = \boldsymbol{D}\hat{\boldsymbol{\mu}}(\boldsymbol{V})$ and $\hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}} = \boldsymbol{D}\hat{\boldsymbol{\Sigma}}(\boldsymbol{V})\boldsymbol{D}^\top$.

Step 2 of the algorithm makes the estimates scale equivariant (by rescaling all the variables), whereas the next steps are a kind principal components that replace the eigenvalues of $\boldsymbol{U}$ (which may be negative definite) by robust

variances. In the OGK algorithm $m(\cdot)$ and $s(\cdot)$ are the τ estimates of location and scale [Yohai and Zamar, 1988] as defaults, but, in our implementation we also offer the option to use the median and the computationally more expensive but location free Qn estimator of scale of [Rousseeuw and Croux, 1993] instead.

3.2.3 The covariance C-step

The covariance C-step was introduced in [Rousseeuw and Van Driessen, 1999] and its general outline is as follow. Consider the MCD objective function [Rousseeuw, 1984, Vakili et al., 2012]:

$$\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}\succeq 0} \frac{1}{p}\log|\boldsymbol{\Sigma}| + \log\sum_{i=1}^{h} d^2(\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\Sigma})_{(i)} \tag{3.7}$$

where $\boldsymbol{A}\succeq 0$ denotes a symmetric positive semi-definite matrix $\boldsymbol{A}$, $\boldsymbol{A}\succ 0$ a symmetric positive definite matrix $\boldsymbol{A}$ and for any $\boldsymbol{A}\succeq 0$, $|\boldsymbol{A}|$ denotes the determinant of $\boldsymbol{A}$ and

$$d^2(\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\Sigma})_i = (\boldsymbol{z}_i-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\boldsymbol{z}_i-\boldsymbol{\mu}). \tag{3.8}$$

The covariance C-step is an iterative procedure: given a pair of initial estimates $(\hat{\boldsymbol{\mu}}^{\text{old}},\hat{\boldsymbol{\Sigma}}^{\text{old}}),\hat{\boldsymbol{\Sigma}}^{\text{old}}\succ 0$, it computes a new pair of estimates $(\hat{\boldsymbol{\mu}}^{\text{new}},\hat{\boldsymbol{\Sigma}}^{\text{new}})$ as follows:

1. Compute the distances $d^2(\boldsymbol{Z},\hat{\boldsymbol{\mu}}^{\text{old}},\hat{\boldsymbol{\Sigma}}^{\text{old}})_i$ for $i=1,\ldots,n$
2. Sort these distances and keep the indexes of the h smallest values of $d^2(\boldsymbol{Z},\hat{\boldsymbol{\mu}}^{\text{old}},\hat{\boldsymbol{\Sigma}}^{\text{old}})_i$ in a subset H:

$$H=\{i: d_i^2(\boldsymbol{Z},\hat{\boldsymbol{\mu}}^{\text{old}},\hat{\boldsymbol{\Sigma}}^{\text{old}})_i \leqslant d^2(\boldsymbol{Z},\hat{\boldsymbol{\mu}}^{\text{old}},\hat{\boldsymbol{\Sigma}}^{\text{old}})_{(h)}\} \tag{3.9}$$

3. Compute $(\hat{\boldsymbol{\mu}}^{\text{new}},\hat{\boldsymbol{\Sigma}}^{\text{new}})$ the maximum likelihood fit of the observations with indexes in H [Dwyer, 1967].

In [Rousseeuw and Van Driessen, 1999], it was proved that

$$|\hat{\boldsymbol{\Sigma}}^{\text{new}}|\leq|\hat{\boldsymbol{\Sigma}}^{\text{old}}| \tag{3.10}$$

with equality only if $\hat{\boldsymbol{\Sigma}}^{\text{new}}=\hat{\boldsymbol{\Sigma}}^{\text{old}}$ so that each covariance C-step decreases the MCD objective function. Therefore, if we apply the covariance C-steps iteratively, the sequence of MCD objective functions obtained in this way must

converge (because for any covariance matrix $\boldsymbol{\Sigma}$ it holds that $|\boldsymbol{\Sigma}| \geq 0$). Since there is no guarantee that the final value of the iteration process is the global minimum of the MCD objective function, an approximate MCD solution is obtained by taking a large number M of initial estimates $\{(\hat{\boldsymbol{\mu}}_m^{\text{init}}, \hat{\boldsymbol{\Sigma}}_m^{\text{init}})\}_{m=1}^M$ applying the covariance C-steps (until convergence) to each, and keeping the h subset H^* yielding the lowest value of the MCD objective function.

3.2.4 The regression C-step

The regression C-step was introduced in [Rousseeuw and Van Driessen, 2006] and its general outline is as follows. Consider the LTS objective function [Rousseeuw, 1984]:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \sum_{i=1}^{h} r_{(i)}^2(\boldsymbol{X}, Y, \boldsymbol{\beta}). \tag{3.11}$$

The regression C-step is an iterative procedure: given an initial estimate $\hat{\boldsymbol{\beta}}^{\text{old}}$, it computes an estimate $\hat{\boldsymbol{\beta}}^{\text{new}}$ as follows:

1. Compute the vector of (residual) distances $r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}})$ for $i = 1, \ldots, n$ (of y_i to $\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}^{\text{old}}$)
2. Sort these distances and keep the indexes of the h smallest values of $r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}})$ in a subset H:
$$H = \{i : r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}}) \leqslant r_{(h)}^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}})\} \tag{3.12}$$
3. Compute $\hat{\boldsymbol{\beta}}^{\text{new}}$ as the LS fit of the observations with indexes in H.

In [Rousseeuw and Van Driessen, 2006], it was proved that

$$\operatorname*{ave}_{i=1}^{h} r_{(i)}^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{new}}) \leqslant \operatorname*{ave}_{i=1}^{h} r_{(i)}^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}}) \tag{3.13}$$

with equality only if $\hat{\boldsymbol{\beta}}^{\text{new}} = \hat{\boldsymbol{\beta}}^{\text{old}}$ so that each regression C-step decreases the LTS objective function. Therefore, if we apply the regression C-steps iteratively, the sequence of LTS objective functions obtained in this way must converge (because $\operatorname*{ave}_{i=1}^{h} r_{(i)}^2(\boldsymbol{X}, Y, \boldsymbol{\beta}) \geq 0$). Since there is no guarantee that the final value of the iteration process is the global minimum of the LTS objective function, an approximate LTS solution is obtained by taking a large number M of initial estimates $\{\hat{\boldsymbol{\beta}}_m^{\text{init}}\}_{m=1}^M$ applying the regression C-step to each, and keeping the h subset H^* yielding the lowest value of the LTS objective function.

3.2.5 Intercept adjustment

The Intercept adjustment step introduced in [Rousseeuw and Van Driessen, 2006] is a technique which decreases the LTS objective function of any fit. After each iteration of the regression C-step we have a vector $\hat{\boldsymbol{\beta}}^{\text{old}}$, yielding an LTS objective value given by Equation (3.11) evaluated at $\hat{\boldsymbol{\beta}}^{\text{old}}$. Denote t_i the univariate set:

$$t_i = y_i - x_{i,1}\hat{\beta}_1^{\text{old}} - \ldots - x_{i,p-1}\hat{\beta}_{p-1}^{\text{old}} \quad \text{for } i = 1, \ldots, n. \tag{3.14}$$

Then, the adjusted intercept $\hat{\beta}_p^{\text{new}}$ is calculated as the exact, univariate LTS location estimate of the t_i, i.e.

$$\hat{\beta}_p^{\text{new}} = \underset{\mu \in \mathbb{R}}{\operatorname{argmin}} \overset{h}{\underset{i=1}{\operatorname{ave}}} (t_i - \mu)^2_{(i)} \tag{3.15}$$

by construction, substituting $\hat{\boldsymbol{\beta}}^{\text{old}}$ by $\hat{\boldsymbol{\beta}}^{\text{new}} = (\hat{\beta}_1^{\text{old}}, \ldots, \hat{\beta}_{p-1}^{\text{old}}, \hat{\beta}_p^{\text{new}})$ always reduces the value of Equation (3.11). Note that Equation (3.15) can be solved by means of an $\mathcal{O}(n \log n)$ algorithm [Rousseeuw and Leroy, 1987, pp 171–172].

3.2.6 I-step

The I-step was introduced in [Salibian-Barrera and Yohai, 2006] and its general outline is as follow. It is a two step iterative procedure that takes as input an initial vector of regression estimates $\hat{\boldsymbol{\beta}}^{\text{old}}$ to compute a new vector of regression estimates $\hat{\boldsymbol{\beta}}^{\text{new}}$. Consider the (regression) S objective function [Rousseeuw and Yohai, 1984]:

$$\begin{aligned} &\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \; s_M \\ s.t. \quad &\overset{n}{\underset{i=1}{\operatorname{ave}}} \; \rho_c \left(r_i(\boldsymbol{X}, Y, \boldsymbol{\beta}) / s_M \right) = b(c) \end{aligned}$$

where $s_M = s_M(\{r_i(\boldsymbol{X}, Y, \boldsymbol{\beta})\}_{i=1}^n)$ with $b(c)$ and c two tuning constants (which we discuss below). Typically, ρ_c is taken to be the Tukey ρ function:

$$\rho_c(u) = \mathcal{I}(|u| \leqslant c) \left(\frac{u^2}{2} - \frac{u^4}{2c^2} + \frac{x^6}{6c^4} \right) + \mathcal{I}(|u| > c) \frac{c^2}{6}$$

where

$$\mathcal{I}(a) = \begin{cases} 1 & \text{if } a \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

and s_M is the univariate S-estimate of scale, e.g. for an n-vector $\boldsymbol{u}$:

$$s_M(\boldsymbol{u}) = \inf\left\{s > 0 : \underset{i=1}{\overset{n}{\text{ave}}}\ \rho(u_i/s) = b(c)\right\}.$$

The derivative of $\rho_c(u)$ w.r.t. to u plays an important role and is known as Tukey's bi-square function and denoted $\psi_c(u)$:

$$\psi_c(u) = \mathcal{I}(|u| \leqslant c)u\left(1 - \frac{u^2}{c^2}\right)^2.$$

The tuning constants c and $b(c)$ are chosen as follows. Under the normal model, [Lopuhaä and Rousseeuw, 1991] showed that the asymptotic breakdown value of the S-estimator is:

$$\varepsilon^* = b(c)/\rho_c(c) \tag{3.16}$$

where, still under the normal model it holds that [Campbell et al., 1998]:

$$b(c) = \frac{1}{2}\chi_3^2(c^2) - \frac{3}{2c^2}\chi_5^2(c^2) + \frac{5}{2c^2}\chi_7^2(c^2) + \frac{c^2}{6}(1 - \chi_1^2(c^2))$$

where χ_d^2 denotes the distribution function of the χ^2 with d degrees of freedom. Then, for a desired breakdown value, one can solve Equation (3.16) iteratively for c. Throughout, we use $\varepsilon^* = 0.5$, yielding $c \approx 1.548$. Then, for a given initial estimate $\hat{\boldsymbol{\beta}}^{\text{old}}$ Each iteration consists of:

1. Compute the (residual) distances $r_i(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}})$ (of y_i to $\boldsymbol{x}_i^\top \hat{\boldsymbol{\beta}}^{\text{old}}$) and the vector of weights w_i with

$$w_i = w\left(r_i(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}}) \Big/ s_M(\{r_i(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{old}})\}_{i=1}^n)\right) \tag{3.17}$$

   where $w(u) = \psi(u)/u$.

2. Compute the resulting $\hat{\boldsymbol{\beta}}^{\text{new}}$, the weighted least squares fit, where observation i is assigned weight w_i:

$$\hat{\boldsymbol{\beta}}^{\text{new}} = (\boldsymbol{X}^\top \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{W} Y \tag{3.18}$$

   where $\boldsymbol{W}$ is a diagonal matrix with diagonal entries $\{w_i\}_{i=1}^n$.

In Lemma 1 of [Salibian-Barrera and Yohai, 2006], it was proved that each I-Step decreases the value of the S objective function so that starting from a vector of initial values for $\hat{\boldsymbol{\beta}}^{\text{old}}$, the I-steps converge. As with the C-step

however, there is no guarantee that the final value of the iteration process is the global minimum of the S objective function. In this case too, typically, practitioners rely on a stochastic approximation to the S solution, obtained by taking a large number M of initial estimates $\{\hat{\boldsymbol{\beta}}_m^{\text{init}}\}_{m=1}^M$ applying the I-step to each, and keeping the solution with lowest objective function.

3.2.7 The M-step

Consider the (regression) MM objective function [Yohai, 1987]:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \underset{i=1}{\overset{n}{\text{ave}}} \rho_c\left(r_i(\boldsymbol{X},Y,\boldsymbol{\beta})/s_0\right)$$

where s_0 is a fixed, high breakdown, scale estimate (typically as obtained after the I-steps have been carried until convergence). The value of c in Equation (3.2.7) determines the asymptotic efficiency of the corresponding estimates. The choice $c = 4.68$ ($c = 3.44$) corresponds to the so-called MM95 (MM85) estimator as it yields an asymptotic efficiency of 95% (85%) for normal errors. For a given initial estimate $\hat{\boldsymbol{\beta}}^{\text{old}}$ (typically as obtained after the I-steps have been carried until convergence) an M-Step iteration is:

1. Compute $r_i(\boldsymbol{X},Y,\hat{\boldsymbol{\beta}}^{\text{old}})$ and set $w_i = w\left(r_i(\boldsymbol{X},Y,\hat{\boldsymbol{\beta}}^{\text{old}})\Big/ s_0\right)$,
2. $\hat{\boldsymbol{\beta}}^{\text{new}}$ is the weighted least square fit where observations i has weight w_i.

It was proved in [Maronna et al., 2006, Section 9] that each M-Step decreases the value of the MM objective function so that starting from a vector of initial values for $\hat{\boldsymbol{\beta}}^{\text{old}}$ the M-steps converge after a finite number of steps (because $\underset{i=1}{\overset{n}{\text{ave}}}\ \rho_c\left(r_i(\boldsymbol{X},Y,\boldsymbol{\beta})/s_0\right) \geq 0$) and, as with the C-steps and I-steps, the MM objective function is not globally convex so that there is no guarantee that this local optimum will also be a global one.

3.2.8 Re-weighting

In order to improve the quality of estimation without compromising the breakdown of the initial estimator, [Rousseeuw and Leroy, 1987] suggested using a weighted least squares (WLS) regression procedure whereby observations with robust standardized residuals larger than some fixed cut-off point are assigned zero weight. Denote

$$r_i^*(\boldsymbol{X},Y,\hat{\boldsymbol{\beta}}) = c_h\ r_i(\boldsymbol{X},Y,\hat{\boldsymbol{\beta}})\Big/\hat{\sigma}(\hat{\boldsymbol{\beta}}) \quad \text{for } i = 1,\ldots,n. \tag{3.19}$$

where $\hat{\sigma}^2(\hat{\boldsymbol{\beta}}) = \underset{i=1}{\overset{h}{\text{ave}}} \; r^2_{(i)}(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}})$ and c_h is a correction factor to obtain consistency when the residuals come from a normal distribution [Pison et al., 2002]. Then, given a vector of parameters $\hat{\boldsymbol{\beta}}^{\text{raw}}$, we define the vector of weights:

$$w_i = \mathcal{I}(r_i^*(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{raw}}) \leqslant \Phi^{-1}(0.9875)) \tag{3.20}$$

Then, the final, re-weighted estimates $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}$ correspond to the weighted least squares fit of the parameters of Model (3.1) to the data with weights w_i.

3.3 The Det family of estimators

The DetS, DetMM and DetLTS algorithms all share a common procedure for the construction of their initial estimates. To obtain these initial estimates, which we denote $\hat{\boldsymbol{\beta}}^{\text{init}}$, we substitute the linear functional in the usual normal equations by a robust alternative based on the OGK estimates of multivariate location and scatter. Below we outlines the approach.

To obtain the initial estimate $\hat{\boldsymbol{\beta}}^{\text{init}}$ we start by applying the OGK algorithm to $\boldsymbol{Z}$. If $\exists \gamma \in \mathbb{R}^p : \gamma^\top \hat{\boldsymbol{\Sigma}}^{\text{OGK}} \gamma = 0$ this means there exists h or more observations lying on a common subspace, constituting a so-called exact fit situation. Then, in those cases, DetLTS returns the indexes of h of these observations, from which the subspace can be recovered. Otherwise, if $\hat{\boldsymbol{\Sigma}}^{\text{OGK}} \succ 0$, we use $(\hat{\boldsymbol{m}}^1, \hat{\boldsymbol{C}}^1) = (\hat{\boldsymbol{\mu}}^{\text{OGK}}, \hat{\boldsymbol{\Sigma}}^{\text{OGK}})$ to compute:

$$\tilde{H}^1 = \{i : d^2(\boldsymbol{Z}, \hat{\boldsymbol{m}}^1, \hat{\boldsymbol{C}}^1)_i \leqslant d^2(\boldsymbol{Z}, \hat{\boldsymbol{m}}^1, \hat{\boldsymbol{C}}^1)_{(h)}\} \tag{3.21}$$

Next, we compute the raw location vector and scatter matrix:

$$\tilde{\boldsymbol{\mu}}^1 = \underset{i \in \tilde{H}^1}{\text{ave}} \, \boldsymbol{z}_i, \quad \tilde{\boldsymbol{\Sigma}}^1 = \underset{i \in \tilde{H}^1}{\text{cov}} \, \boldsymbol{z}_i \tag{3.22}$$

Next, we use $\tilde{\boldsymbol{\mu}}^1$ and $\tilde{\boldsymbol{\Sigma}}^1$ as starting points for the covariance C-step algorithm, run it on $\boldsymbol{Z}$ until convergence, yielding the new estimates $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ and, if $\hat{\boldsymbol{\Sigma}}^1 \succ 0$, the set H^1:

$$H^1 = \{i : d^2(\boldsymbol{Z}, \hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)_i \leqslant d^2(\boldsymbol{Z}, \hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)_{(h)}\} \tag{3.23}$$

Finally, we obtain $\hat{\boldsymbol{\beta}}^{\text{init}}$ by a least squares fit of Model (3.1) to the observations with indexes in H^1. From this point on, the procedures used to obtain the three "Det" algorithms differ.

For DetLTS, from this initial estimate $\hat{\boldsymbol{\beta}}^{\text{init}}$, we carry out regression C-steps (with intercept adjustment) until convergence, yielding $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}$ and $\hat{\sigma}^{\text{DetLTS}}$. Finally, we carry out a one step re-weighting on $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}$ and $\hat{\sigma}^{\text{DetLTS}}$ yielding $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}$ and $\hat{\sigma}^{\text{DetLTS}}$. If we substitute the regression C-steps in the procedure above by I-steps (again, carried until convergence), we call the resulting estimator DetS and denote the resulting estimates $\hat{\boldsymbol{\beta}}^{\text{DetS}}$ and $\hat{\sigma}^{\text{DetS}}$. Finally, to obtain the DetMM estimates of the parameters in Model (3.1), we simply carry the M-steps from $\hat{\boldsymbol{\beta}}^{\text{DetS}}$ and $\hat{\sigma}^{\text{DetS}}$ until convergence, yielding $\hat{\boldsymbol{\beta}}^{\text{DetMM}}$ and $\hat{\sigma}^{\text{DetS}}$ (the M-steps do not change the initial estimate of scale).

The FastLTS, FastS and FastMM algorithms are affine and regression equivariant. In contrast, due to the construction of their initial estimates, $\hat{\boldsymbol{\beta}}^{\text{init}}$, DetS, DetLTS and DetMM are not affine or regression equivariant. Likewise that FastLTS, FastS and FastMM are not permutation invariant because the initial subsets (which are generated by a pseudo-random number generator with a fixed seed) will have the same case numbers but correspond to different observations. By contrast, all ingredients of DetLTS, DetS and DetMM are permutation invariant.

To enhance user experience, we implemented DetMM, DetS and DetLTS in a portable `R` package (package `DetR`). For computational efficiency, the initial step used to obtain $\hat{\boldsymbol{\beta}}^{\text{init}}$ as well as the C-steps used in the DetLTS algorithm have been implemented in C++using modern, state-of-the-art numerical libraries [Guennebaud et al., 2013] with an emphasize on performance. For the I-step used in the computation of the DetS algorithm, we used a lightly modified version of the `R` implementation of [Salibian-Barrera and Yohai, 2006] available on the author's website (`http://www.stat.ubc.ca/~matias/soft.html`). Finally, for the M-step algorithm used in the computations of the DetMM algorithm, we used the `lmrob` function available in the `R` package `robustbase`[Rousseeuw et al., 2014] (available on `CRAN`). Moreover, the `DetR` package also contains a `R` implementation of the initial step used to obtain $\hat{\boldsymbol{\beta}}^{\text{init}}$ in as the C-steps and a series of test functions used to ensure correctness of and illustrate various parts of the computations. The help file of the `test_function`, which can be accessed by typing:

```
R> library("DetR")
R> ?test_function
```

into the R prompt, contains documented and illustrated examples to enable the reader to do this. To simplify the reproduction of the tests and comparisons, the DetR package also contains the data set used in the following section.

3.3.1 Running Times

In this section we compare the running times of the deterministic and random algorithms. Specifically, we will focus on FastLTS and DetLTS since they are the fastest representatives of their respective groups. In practice, (computational) resources are never unbounded so that running times is always an important characteristic of an algorithm. Moreover, like some of their counterparts in the context of robust estimation of covariance [Billor et al., 2000], [Maronna and Zamar, 2002] the algorithms we present in this chapter are explicitly geared towards higher dimensional problems (often defined as p larger than 20 say). Since the numerical complexity of robust algorithm typically does not grow linearly in p, this makes running times even more important. For DetLTS, we used our own implementation whereas for FastLTS we used a state-of-the-art implementation: the ltsReg function in the R package robustbase [Rousseeuw et al., 2014] with default options.

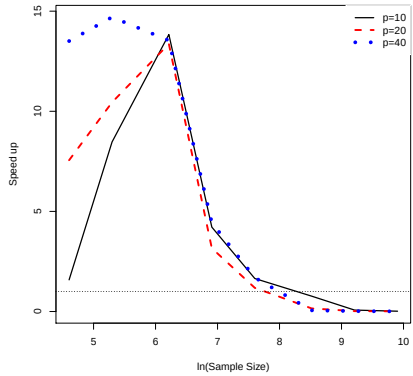


Figure 3.1: Speed up of DetLTS over FastLTS for different values of p and n. The vertical axis shows the (harmonic) mean of the relative running times for the corresponding values of n and p. Values larger than one indicate that DetLTS is on average quicker than FastLTS for the corresponding value of n and p.

When it is run with the τ estimator of scale, the numerical complexity of DetLTS scales as $O(np^2)$. This is because the most time consuming part of the algorithm is the computation of the G.K. covariance matrix which involves $O(p^2)$ calls to the τ estimator of scale (each at cost $O(n)$). When $n \geq 600$ FastLTS uses a nested sub-sampling scheme whereby larger datasets are divided into non-overlapping sub-samples of at most 600 observations. Furthermore, the FastLTS computes two C-steps and retains only the 10 subsets with lowest value of the objective function for further refinement (on these, the C-steps are carried until convergence). Because the sub-sampling algorithm has a random component, it cannot be used by DetLTS to speed up the computations. Therefore, for large values of n we do not expect DetLTS to be faster than FastLTS. In Figure 3.1 we depict the average speed up of DetLTS over FastLTS for different values of n and p for $n = \{100, 200, 500, 1000, 2000, 5000, 10000, 20000\}$ and $p = \{10, 20, 40\}$. The speed up on the i-th experiment is defined as:

$$\text{Speed-up(FastLTS,DetLTS)}_i = \frac{\text{Time(FastLTS)}_i}{\text{Time(DetLTS)}_i} \tag{3.24}$$

For each combination of n and p we run ninety-six experiments and take the harmonic mean of the vector of (96) speed up values. By construction, a value of 1 for a particular setting indicates that both algorithm take on average a similar amount of time to find a fit on that setting. As expected, the most important speed ups are obtained when $\log(n)$ is just above 6 (corresponding to $n = 500$) and are between a factor of 10 and 20 (with the larger multiples corresponding to the smaller values of p), since the nested sub-sampling scheme used by FastLTS kicks in for larger values of n. From there, the speed ups decrease as n becomes larger and eventually, at around $n = 3000$ FastLTS becomes faster than DetLTS. It would have been interesting to know whether the number of regression C-steps needed to insure convergence for FastLTS is lower than for DetLTS but, unfortunately, the implementation of `ltsReg` we used does not output this information.

3.3.2 Different Values of h

For n smaller than approximately 3000, the deterministic algorithms are faster than their "Fast" counterparts when applying the algorithm for a fixed value of h. In situations where the actual number of outliers is expected to be below $n/2$, it is commonly advised to use higher values of h such as $h \approx 3n/4$ to improve the efficiency of the fit [Rousseeuw et al., 1999b]. In situations where the outliers are well separated from the good observations, one could search for the largest value of h smaller than the contamination rate of the sample.

In practice, this could be done by computing the LTS, S and MM objective functions for many values of h and see whether an important change in the objective function or the estimates occurs at some value of h. This approach is related to the forward search strategy adopted in [Atkinson et al., 2004], though a similar diagnostic tool has been used in the context of univariate robust fitting for a long time [Rosenberger and Gasko, 1983].

The new deterministic algorithms we introduced above render the task of computing the LTS, S and MM objective function for a grid of values of h much faster. This is because the initial estimates do not depend on h so that the algorithm only needs to store the vector of squared residuals corresponding to the unique initial estimate $\hat{\boldsymbol{\beta}}^{\text{init}}$. Then, from this, we get the final h-subset for any h by running C, I and M steps up to convergence for a grid of values of h. We will illustrate this for DetLTS in the real data example of Section 3.4.

3.4 Illustration on a real data example

In this Section, the focus is again on comparing the Det algorithms with their Fast counterparts, but this time we carry out the comparisons on a large, multi-variable real data example. To reduce duplications, we focus here on two algorithms, FastLTS and DetLTS. More precisely, our dataset consists of the adult component of the 2009 California Health Interview Survey (`www.chis.ucla.edu`). The CHIS is a population-based telephone survey of California's population. The survey aims to collect extensive information on health status, health conditions, health-related behaviors, health insurance coverage as well as access to health care services. Within each household, separate interviews are conducted with a randomly selected adult (age 18 and over). The dataset consists of 536 features measured for 47614 respondents.

For this exercise, our purpose is to model the weight (in kilograms) of the surveyed individuals as a function of a set of covariates. More precisely, we will focus on the subsample of 17179 individuals with age greater than 29. As response variable, we use the weight (measured in kilograms) at the time of the survey.

We begin by removing the design variables having 0 values of the MAD (for example the variable 'age when period started' has value -1 –coding convention for non 'NA'– for all observations in our subsample) as well as the variables that are causing exact fit on the design space (for example, most measurements are recorded in both ISI and imperial units. In those cases we always retain the ISI values). The ordinal variable "educational attainment" has been recoded to their cumulative proportions within the sample (for example, all observations

for which "educational attainment" is equal to "grade 9-11" have been recoded as 0.1015 since this is the proportion of the sample with values of "educational attainment" equal to "grade 9-11" and lower.) We also removed some variables that have no information content (for example the variable "name of health plan"). All told, we have 25 variables left.

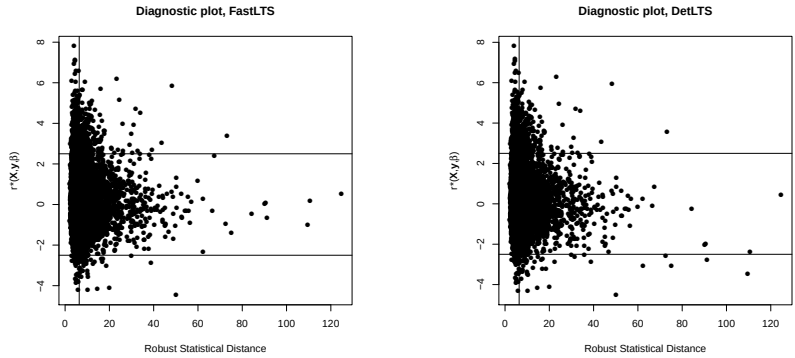


Figure 3.2: Regression diagnostic plots obtained on the CHIS 2009 dataset. Left for FastLTS and right for DetLTS.

We start by running FastLTS and DetLTS on this dataset. Both algorithms have a parameter $\alpha \in [0.5, 1]$ which controls the size of the subset over which the LTS objective function is computed through $h \approx [\alpha \times n]$. In both cases, we set α to 0.5 to get the most robust fits (for FastLTS we also fix the seed to ensure easy replication and set the number of starting subsets to 500). The two functions have very close values of the LTS objective function (10.81 vs 10.77) though that of DetLTS is slightly smaller. In both cases, the proportion of observations awarded a weight in the final (re-weighted fit) is high (about 94%), indicating that only a a small proportion of the observations lie really far from the fit.

The (residual) outliers map is a diagnostic tool for robust regression [Rousseeuw and Van Zomeren, 1990]. This diagnostic tool displays the standardized residuals obtained from a robust fit versus robust statistical distances (computed on the design variables). Two horizontal lines are located at +2.5 and -2.5 and the vertical line is located at the upper 0.975 percent point of the chi-squared distribution with $p-1$ degrees of freedom. Observations falling outside of the rectangle delimited by the two horizontal and to the left of the line $x = \sqrt{\chi^2_{p-1}(0.975)}$ are identified as vertical outliers. Observations falling inside

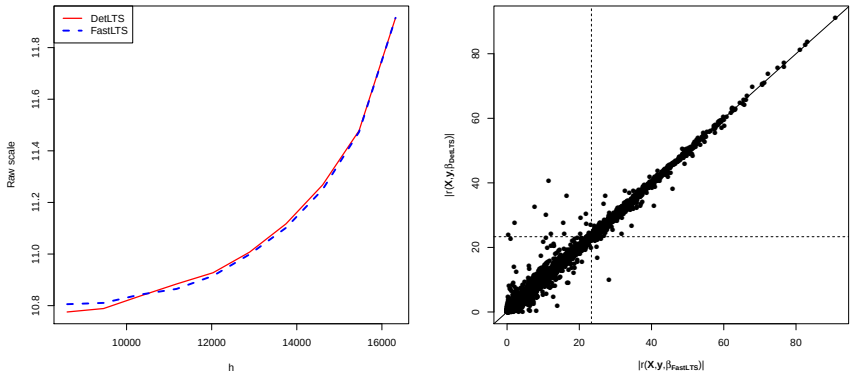


Figure 3.3: Value of the raw scales, as a function of h for FastLTS (dashed line) and DetLTS (full line) for the CHIS 2009 data-set (left). Residual-Residual (distance) plot of FastLTS versus DetLTS (right) corresponding to the solution with $\alpha = 0.5$.

the rectangle delimited by the lines and $y = \pm 2.5$ and to the right of the line $x = \sqrt{\chi^2_{p-1}(0.975)}$ are identified as good leverage outliers. Finally, the observations falling outside $y = \pm 2.5$ and to the right of $x = \sqrt{\chi^2_{p-1}(0.975)}$ are identified as bad leverage outliers.

The outliers maps derived from the two fits are shown in Figure 3.2 (to make the results more easily comparable, the robust distances depicted on the horizontal axis of the outliers maps are identical for both estimators) and are very similar, thought DetLTS flags slightly less bad leverage outliers (335 vs 340). The left plot in Figure 3.3 depicts the values of the DetLTS and FastLTS raw scales as a function of h for 10 values of α equidistant between 0.5 to 0.95. The lines in the left plot of Figure 3.3 show no obvious break and are therefore hard to interpret. They are consistent with the plot of the residuals from both fits depicted in the right of Figure 3.3 which, though it shows a large contingent of observations with abnormally large residuals, fails to reveal a clearly identifiable cluster demarcated from the main cluster of observations (as identified by both DetLTS and FastLTS). In other words, neither the FastLTS nor the DetLTS fits reveals any outlying cluster of observations clearly separated from the bulk of the data. However, for $\alpha = 0.5$ (corresponding to the more robust estimates), the fit found by DetLTS seems to adjust the observations more tightly. The right plot in Figure 3.3 shows the plot of the residuals of both fits, together with horizontal lines indicating twice the estimated scales. For comparison, producing the left

plot of Figure 3.3 took about four times more time for FastLTS than DetLTS.

A simple way to compare how well the two robust estimators fit the uncontaminated observations in a given data set is to consider the members of the subset:

$$H^{+} = \{H^{\text{FastLTS}} \cap H^{\text{DetLTS}}\}$$

where

$$\begin{aligned} H^{\text{FastLTS}} &= \{i : r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{FastLTS}}) \leqslant r_{(h)}^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{FastLTS}})\} \\ H^{\text{DetLTS}} &= \{i : r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{DetLTS}}) \leqslant r_{(h)}^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{DetLTS}})\} \end{aligned}$$

Since $h > n/2$, $\#\{H^{+}\} \geq p+1$. Then, if either one of H^{FastLTS} H^{DetLTS} is free of outliers, so is H^{+}. Next we consider the statistics:

$$d_h = \underset{i \in H^{+}}{\text{ave}} (r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{FastLTS}}) - r_i^2(\boldsymbol{X}, Y, \hat{\boldsymbol{\beta}}^{\text{DetLTS}}))$$

to determine which one of $\hat{\boldsymbol{\beta}}^{\text{FastLTS}}$ or $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}$ fits the uncontaminated observations better. In this example, for $\alpha = 0.5$, we find that $d_h > 0$ (with $\#\{H^{+}\} = 6438$) so that, at least for the most robust versions of the solutions, we find evidence that $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}$ fits the good part of the data better than $\hat{\boldsymbol{\beta}}^{\text{FastLTS}}$.

One can also compare the two robust linear regression fits in terms of their estimated coefficients (and the estimates of standard errors corresponding the FastLTS fit), shown in Table 3.6 (presented at the end of this chapter). Nonetheless, we show the fitted coefficients as well as their t-statistics in Table 3.6 for illustration's sake. The fitted coefficients from the two approach are generally close to one another, confirming the results from the other comparisons based on residuals, that the two estimators find very similar fits on this dataset.

3.5 Discussion

One could contemplate replacing $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ in Equation (3.28) by $(\hat{\boldsymbol{\mu}}^{\text{DetMCD}}, \hat{\boldsymbol{\Sigma}}^{\text{DetMCD}})$, the DetMCD estimates of location and scatter [Hubert et al., 2012] (a similar argument could be made for the DetS estimates of location and scatter [Hubert et al., 2015c]). Given $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ and 5 other

deterministic candidate fits $(\hat{\boldsymbol{\mu}}^j, \hat{\boldsymbol{\Sigma}}^j)|j = 2, \ldots, 6$ (and the corresponding h subsets $H^j|j = 2, \ldots, 6$), the DetMCD estimates of location and scatter correspond to the candidate fit having smallest value of $|\hat{\boldsymbol{\Sigma}}^j|$. Like $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$, the DetMCD solution is also deterministic. However, compared to $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$, $(\hat{\boldsymbol{\mu}}^{\text{DetMCD}}, \hat{\boldsymbol{\Sigma}}^{\text{DetMCD}})$ suffers from three important disadvantages which are discussed below and motivate our choice in favor of the former. In the sequel, DetMCDLTS will be the algorithm obtained by replacing $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ by $(\hat{\boldsymbol{\mu}}^{\text{DetMCD}}, \hat{\boldsymbol{\Sigma}}^{\text{DetMCD}})$ in the computation of DetLTS. Some of the arguments are illustrated by numerical comparison. In all cases, those comparisons were carried using the `robustbase` [Rousseeuw et al., 2014] implementation of DetMCD with default options except for the use of the τ estimates of location and scale (for the values of n considered here, the default scale used in the computations of DetMCD is the Q_n). In particular, the rest of the code base for DetMCDLTS is the same as that of DetLTS.

The first argument is that increasing the number of candidate fits (from one to six) inevitably increases the computational cost of obtaining the initial estimates of location and scatter of $\boldsymbol{Z}$. To illustrate how large the effect is, it is enough to re-run the experiments of Section 3.3.1 but this time using DetMCDLTS and compare the results with those obtained in Section 3.3.1.

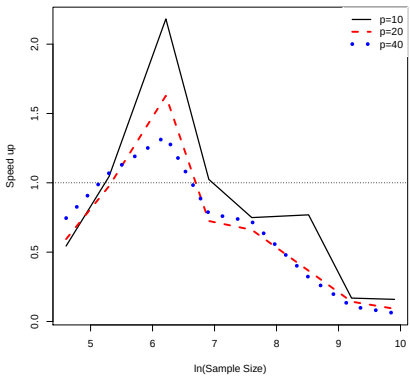


Figure 3.4: Speed up of DetMCDLTS over FastLTS for different values of p and n.

As shown in Figure 3.4 the use of many alternative starts is very demanding computationally. Focusing on the situation when p is large (computational efficiency matters most when the size of the problem is larger) we see that,

as with DetLTS, the relative speed up of DetMCDLTS increases with n until about $n = 600$ at which point sub-sampling yields increasingly larger gains for FastLTS, enabling it to eventually out-pace DetMCDLTS. However, compared to DetLTS, the window of values of n for which DetMCDLTS is faster than FastLTS has shrunk considerably. Whereas DetLTS runs faster than FastLTS for up to $n = 3000$, for DetMCDLTS, the cross-over occurs at about $n = 700$. Furthermore, the gains in computational times are also much smaller. For example, when $n = 500$, DetLTS is about 14 times faster than FastLTS whereas DetMCDLTS is only 35% faster. Considering only those values of n for which DetMCDLTS runs faster than FastLTS, one finds that DetLTS runs about an order of magnitude faster than the former.

The second argument is that in some situations we find that $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ is less affected by the outliers than DetMCD. This means that in those situations, the MCD objective function systematically chooses a sub-optimal candidate among the 6 starts used in DetMCD. To illustrate this, consider the following modest simulation experiment. We generated $M = 100$ contaminated dataset $\boldsymbol{X}^\varepsilon = ((\boldsymbol{X}^u)^\top, (\boldsymbol{X}^c)^\top)^\top$ where the uncontaminated data is $\boldsymbol{X}^u$ with $\boldsymbol{X}^u \sim \mathcal{N}(\boldsymbol{0}_{p-1}, \boldsymbol{\Sigma}^u)$ where $\boldsymbol{\Sigma}^u$ is a correlation matrix with out of diagonal entries chosen such that the squared multiple correlation between the first $p-1$ columns of $\boldsymbol{X}^u$ is $\rho = 0.95$, the outliers $\{\boldsymbol{x}_i^c\}_{i=1}^{\lfloor n\varepsilon \rfloor} = \boldsymbol{X}^c$ are of the Point-mass variety placed at distance $d_{\boldsymbol{X}} = 10$ from the uncontaminated observations, where:

$$\min_i \left(\mathrm{E}(\boldsymbol{X}^u) - \boldsymbol{x}_i^c\right)^\top (\boldsymbol{\Sigma}^u)^{-1} \left(\mathrm{E}(\boldsymbol{X}^u) - \boldsymbol{x}_i^c\right) = d_{\boldsymbol{X}}^2 \chi^2_{p-1;0.975} \tag{3.25}$$

We then compare two estimators of $\boldsymbol{\Sigma}^u$ in terms of their biases, measured as in [Maronna and Zamar, 2002]. The first estimator (denoted "DetMCD") is the DetMCD estimator [Hubert et al., 2012], using the `robustbase`[Rousseeuw et al., 2014] implementation of the algorithm with default settings. The second estimator (denoted "OGKCStep") is $\hat{\boldsymbol{\Sigma}}^1$, but computed using the Qn estimator of scale of [Rousseeuw and Croux, 1993] as explained in section 3.2.2 (this is to make the results more comparable since by default, DetMCD uses the Qn estimator of scale). As the results in Table 3.1 show, for high rates of contamination, the estimator based on 6 starts (DetMCD) under-performs the estimate based on a single start $\hat{\boldsymbol{\Sigma}}^1$. To facilitate reproducibility a stand alone version of the code used to obtain Table 3.1 is included in the `DetR` package (together with an example and documentation) and can be accessed by typing:

```
R> library("DetR")
R> ?OGKCStep
```

in the R prompt.

	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.3$	$\varepsilon = 0.4$
OGKCStep	0.49	0.44	0.38	0.26
DetMCD	0.47	0.43	4.06	4.63

Table 3.1: Bias due to bad leverage Point mass outliers. Correlated Gaussians, $p = 40$, $d_{\boldsymbol{X}} = 10$.

Since neither estimator is affine equivariant, the biases of both will be affected by the covariance structure of the uncontaminated observations as well as the configuration of the outliers. This problem also affects the simulations carried in Chapter 2. In particular, the lower biases of DetMCD compared to OGK reported there in some settings are highly contingent on the covariance structure of the uncontaminated data. More precisely, the simulation settings used in Chapter 3 have higher values of ρ than those of Chapter 2. All other things equal, higher values of ρ are known to negatively (and dramatically) affect the behavior of non affine equivariant algorithms and at the limit even cause their biases to become arbitrarily large ([Maronna and Zamar, 2002]). Given this, it is perhaps not surprising that the relative performance of the two algorithms is also, at least in the selected settings we considered, severely affected by the choices of the values of ρ (another, less important factor is that DetLTS uses a combination of one step OGK and CSteps as opposed to the two step OGK approach used in Chapter 2). Furthermore, to the best of our knowledge, the configuration of outliers adversary for shift and scale equivariant estimators are not known. Therefore, any argument about the relative robustness of these estimators based on simulations are to a great extent speculative. Nonetheless, the modest simulation above shows that it is not at all guaranteed that the more complicated DetMCD always does better in terms of bias than the nimbler OGKCStep.

The situation becomes even more complicated in the case of the non regression and affine equivariant estimators of regression we consider in greater detail here. Now, the relative performance of the estimators depend on the covariance of the good data and the regression hyperplane fitting them as well as the configuration of the outliers. Furthermore, to the best of our knowledge there no arguments as to what configuration of outliers, if any, is in some sense adversary for shift and scale equivariant estimators of regression. In fact, because the exact fit property has not been established for either DetMCDLTS or DetLTS, the maximum bias of both may well be unbounded in situations where the uncontaminated observations lie on an subspace. For all these reasons, the results of the modest simulation study we present below should be interpreted with caution. Nonetheless, to compare DetMCDLTS to DetLTS quantitatively, we again

generated $M = 100$ contaminated dataset $(\boldsymbol{X}^{\varepsilon}, Y^{\varepsilon})$ where $\boldsymbol{X}^{\varepsilon}$ is defined as above and $Y^{\varepsilon} = ((Y^u)^\top, (Y^c)^\top)^\top$ where $Y^u = \{y_i^u\}_{i=1}^{\lceil n(1-\varepsilon)\rceil}$ and $Y^c = \{y_i^c\}_{i=1}^{\lfloor n\varepsilon\rfloor}$ with

$$\begin{aligned} y_i^u &= (\boldsymbol{x}_i^u)^\top \boldsymbol{\beta} + \epsilon_i \\ y_i^c &= (\boldsymbol{x}_i^c)^\top \boldsymbol{\beta} + 0.01\epsilon_i - d_y \end{aligned}$$

$\epsilon_i \sim$ i.i.d. $\mathcal{N}(0,1)$, $\boldsymbol{\beta} = \mathbf{1}_p$ and $d_y = 10$. This setting corresponds to so called bad leverage outliers which are are data points that stand out from the genuine observations both on the design space and in terms of their distances to the regression hyperplane fitting them [Rousseeuw and Van Zomeren, 1990]. We also include three popular robust regression estimators to these comparison (FastLTS, FastS, FastMM, all three from `robustbase`[Rousseeuw et al., 2014] and used with their default settings). As in Chapter 2, we find that in a small number of cases (in as much as 5% of the cases) FastS and FastMM do not converge. When this occurred, we disregarded those individual results. We then compare the two estimators of $\boldsymbol{\beta}$ in terms of their biases, measured as: $\text{ave}_{m=1}^{100} ||\hat{\boldsymbol{\beta}}_m^{\text{est}} - \boldsymbol{\beta}||$ where est $= \{\text{DetLTS}, \text{DetMCDLTS}, \text{FastLTS}, \text{FastS}, \text{FastMM}\}$. The code to reproduce the results shown in Table 3.5 below (as well as those shown in Table 3.4) is included (documented and illustrated) in the `DetR` package and can be accessed by typing:

```
R> library("DetR")
R> ?test_function_bias
```

Using the code above, we find again that at least in this setting, the more complicated DetMCDLTS algorithm does not outperform the nimbler DetLTS.

A second popular scenario is the one corresponding to so-called vertical outliers

	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.3$	$\varepsilon = 0.4$
DetLTS	1.17	1.16	2.58	3.63
DetMCDLTS	1.20	1.12	2.43	3.65
FastLTS	1.50	2.12	3.04	4.40
FastS	2.12	2.32	2.40	2.84
FastMM	1.04	1.26	2.15	2.89

Table 3.2: Bias due to bad leverage outliers, $d_y = 10$, $p = 40$.

which are are data points that stand out from the genuine observations only in terms of their distances to the regression hyperplane fitting them [Rousseeuw

and Van Zomeren, 1990]. In our framework, one example of this configuration of outliers is obtained by generating $M = 100$ contaminated dataset $(\boldsymbol{X}^{\varepsilon}, Y^{\varepsilon})$ as above but this time setting $d_{\boldsymbol{X}} = 0$. As before, we find that in this second setting as well, the more complicated DetMCDLTS algorithm does not outperform the nimbler DetLTS.

	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.3$	$\varepsilon = 0.4$
DetLTS	1.16	1.08	32.39	41.31
DetMCDLTS	1.16	1.08	33.01	42.43
FastLTS	0.98	0.98	28.12	47.02
FastS	1.60	1.39	38.10	50.90
FastMM	0.90	0.96	3.94	46.03

Table 3.3: Bias due to vertical outliers, $d_y = 10$, $p = 40$.

Below, we also repeated the simulations of Tables 3.2 and 3.1, but this time setting $p = 10$ (keeping all other simulation parameters fixed).

	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.3$	$\varepsilon = 0.4$
DetLTS	0.49	0.48	23.65	33.98
DetMCDLTS	0.49	0.49	23.58	35.33
FastLTS	0.47	0.47	0.51	37.80
FastS	0.73	0.66	0.58	22.74
FastMM	0.44	0.47	0.51	11.26

Table 3.4: Bias due to vertical outliers, $d_y = 10$, $p = 10$.

	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.3$	$\varepsilon = 0.4$
DetLTS	0.51	0.51	0.55	2.33
DetMCDLTS	0.51	0.52	1.61	2.40
FastLTS	1.00	1.28	1.59	2.22
FastS	1.24	1.44	1.66	1.65
FastMM	0.87	0.90	1.00	1.57

Table 3.5: Bias due to bad leverage outliers, $d_y = 10$, $p = 10$.

The third argument in favor of using $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ instead of $(\hat{\boldsymbol{\mu}}^{\mathrm{DetMCD}}, \hat{\boldsymbol{\Sigma}}^{\mathrm{DetMCD}})$ is related to the derivation of the finite sample breakdown of the resulting regression estimator. Loosely speaking, to measure the robustness of an estimator to the presence of outliers in the data we often use the notion of finite sample breakdown point of an estimator, as introduced by [Donoho, 1982]. Given a sample and an estimator, this is the smallest proportion of observations that needs to be replaced by arbitrary values to cause the fitted coefficients to take on values on the boundary of the parameter space. Remarkably, the finite sample breakdown

point of an estimator can be derived without recourse to concepts of chance or randomness using geometrical features of a sample and the estimator alone. More formally, write the contaminated sample as $\boldsymbol{Z}^m$. This contaminated sample is obtained by replacing m observations of $\boldsymbol{Z}$ by arbitrary values. Then, the definition of finite sample breakdown point of a regression estimator $\hat{\boldsymbol{\beta}}$ at the sample $\boldsymbol{Z}$ (formed of the first $p-1$ columns $\boldsymbol{X}$ and Y) is adapted from [Maronna et al., 2006, p. 122]:

$$\varepsilon_n^*(\hat{\boldsymbol{\beta}}, \boldsymbol{Z}) = \min\left\{\frac{m}{n} : \sup_{\boldsymbol{Z}^m} ||\hat{\boldsymbol{\beta}}(\boldsymbol{Z}^m)||_2 = \infty\right\}. \tag{3.26}$$

and $\boldsymbol{Z}_m$ is *any* data sets with at least $m-n$ elements in common with $\boldsymbol{Z}$. We also assume throughout that the observations in $\mathbb{Z}$ are in general position in $\mathbb{R}^p$. The rows of an n by p data matrix $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$ if no more than p points of $\boldsymbol{Z}$ lie in any $p-1$-dimensional affine subspace [Rousseeuw and Leroy, 1987, p. 257]. We will also assume that $n > p$. These assumptions (as in for example [Tyler, 1994]) all pertain to the original, uncontaminated, data set $\boldsymbol{Z}$. Under these assumptions, and denoting $\lfloor a \rfloor$ the largest integer not greater than x and $\lceil a \rceil$ the smallest integer not less than x, we show in Appendix 1 that

Theorem 1. *Let $\boldsymbol{Z}$ be a sample in general position in $\mathbb{R}^p$. Let h be an integer satisfying $n \geqslant h \geqslant \lceil \frac{n+p+1}{2} \rceil$, then*

$$\varepsilon_n^*(\hat{\boldsymbol{\beta}}^{DetLTS}, \boldsymbol{Z}) \geqslant \frac{(n-h+1)}{n}.$$

By taking $h = \lceil \frac{n+p+1}{2} \rceil$ we get a (finite sample) breakdown value of $\lceil \frac{n-p}{2} \rceil$.

As we show in Appendix 1 below, the finite sample breakdown point of the DetR estimators can be derived pretty straightforwardly from that of $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$, the initial location and scatter estimates used to obtain $\hat{\boldsymbol{\beta}}^{\text{init}}$. However, because the location and scatter (finite sample) breakdown point of the DetMCD [Hubert et al., 2012] or DetS [Hubert et al., 2015c] estimators has not yet been published, adopting either one of $(\hat{\boldsymbol{\mu}}^{\text{DetMCD}}, \hat{\boldsymbol{\Sigma}}^{\text{DetMCD}})$ $(\hat{\boldsymbol{\mu}}^{\text{DetS}}, \hat{\boldsymbol{\Sigma}}^{\text{DetS}})$ in place of $(\hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)$ in the first stage of the DetR algorithms would complicate the derivation of the (regression) breakdown point of the DetR estimators.

3.6 Concluding Remarks

To the best of our knowledge, positive breakdown regression estimators with computational complexities allowing them to be run on large data are either

affine equivariant or permutation invariant. Historically, the design choice for statisticians has usually been to sacrifice the latter property in order to preserve the former. Looking back, it is important to recall that this choice was in no small part guided by the fact that equivariance greatly simplifies the derivation of key theoretical properties of an estimator. However, in the presence of contaminated data-sets (that is, when convex stress functions do not accurately reflect our preferences), it is not at all clear that equivariant estimators systematically yield better outcomes than permutation invariant ones [Lehmann and Casella, 2003, Theorem 9.2]. In this regard, the present chapter (along with [Hubert et al., 2012] and [Hubert et al., 2015c]) can be seen as an exercise in exploring the consequences, of choosing the alternative venue –e.g of choosing permutation invariance over affine equivariance.

To conclude, we note that, as with the earlier related DetMCD [Hubert et al., 2012] and DetS [Hubert et al., 2015c] algorithms, the statistical consistency of the DetR family of methods as not as of yet been established and that doing so would be desirable.

The author is grateful to Viktoria Öllerer for her constructive comments on a draft of Appendix 1 and to Eric Schmitt for proof reading an earlier version of this text for typographical errors. The author is also indebted to Christophe Croux for his many advices, guidances and review of the final draft of Appendix 1. All remaining mistakes are of course mine.

Appendix 1: Lower bound on the finite sample breakdown point of $\hat{\beta}^{\mathsf{DetLTS}}$.

Proof of Theorem 1. Take a contaminated sample $\boldsymbol{Z}^m$ obtained by replacing m of the n entries of $\boldsymbol{Z}$ by arbitrary values with $m \leqslant n-h$. To establish that the finite sample breakdown point of $\hat{\boldsymbol{\beta}}^{\mathrm{DetLTS}}$ is at least as large as $\left\lceil \frac{n-p}{2} \right\rceil$, it is sufficient to prove that there exist a positive constant N depending only on the original observations $\boldsymbol{Z}$ such that

$$\left|\left|\hat{\boldsymbol{\beta}}^{\mathrm{DetLTS}}(\boldsymbol{Z}^m)\right|\right|_2 \leqslant N$$

whenever the entries of $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$. Denote $(\hat{\boldsymbol{\mu}}^{\mathrm{OGK}}(\boldsymbol{Z}^m), \hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m))$ the OGK estimator of location and scatter computed at $\boldsymbol{Z}^m$ and (t,s) the univariate estimates of location and scatter used to compute $(\hat{\boldsymbol{\mu}}^{\mathrm{OGK}}(\boldsymbol{Z}^m), \hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m))$. It has already been shown (under the condition that the entries of $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$) that the finite sample breakdown point of $(\hat{\boldsymbol{\mu}}^{\mathrm{OGK}}, \hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}})$ is as high as that of (t,s) [Maronna and Zamar, 2002] which, in the case of the DetR family of estimator, is at least $(n-h+1)/n$. This means that there exists constants $\delta_1 > 0$, $M_1 > 0$ and $M_2 > 0$ depending only on the uncontaminated sample $\boldsymbol{Z}$ such that

$$0 < \delta_1 \leqslant |\hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m)| \leqslant M_1 < \infty, \tag{3.27}$$

and

$$||\hat{\boldsymbol{\mu}}^{\mathrm{OGK}}(\boldsymbol{Z}^m)||_2 \leqslant M_2 < \infty.$$

Denote

$$\tilde{H}^1 = \{i : \quad d^2(\boldsymbol{Z}^m, \hat{\boldsymbol{\mu}}^{\mathrm{OGK}}(\boldsymbol{Z}^m), \hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m))_i \leqslant$$

$$d^2(\boldsymbol{Z}^m, \hat{\boldsymbol{\mu}}^{\mathrm{OGK}}(\boldsymbol{Z}^m), \hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m))_{(h)}\}$$

such that

$$\tilde{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m) = \underset{i \in \tilde{H}^1}{\mathrm{ave}}\, \boldsymbol{z}_i^m, \quad \tilde{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m) = \underset{i \in \tilde{H}^1}{\mathrm{cov}}\, \boldsymbol{z}_i^m.$$

Recall that $\tilde{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m)$ and $\tilde{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)$ are used as starting points for the covariance C-step algorithm, which is run on $\boldsymbol{Z}^m$ until convergence, yielding the new estimates $(\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m), \hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m))$ and the set H^1:

$$H^1 = \{i : d^2(\boldsymbol{Z}, \hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)_i \leqslant d^2(\boldsymbol{Z}, \hat{\boldsymbol{\mu}}^1, \hat{\boldsymbol{\Sigma}}^1)_{(h)}\} \tag{3.28}$$

Denote $G^1 = H^1 \cap I^G$ where I^G is the set of indexes of the uncontaminated rows of $\boldsymbol{Z}^m$. Since $\#G^1 \geqslant h - n \geqslant 2h - n \geqslant n + p + 1 - n \geqslant p + 1$, H^1 contains at least $p + 1$ observations in general position in $\mathbb{R}^p$. Since the entries of $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$, there exists a strictly positive constant δ_2 depending only on the entries of $\boldsymbol{Z}$ such that

$$\inf_{\substack{\#G \geqslant p+1, \\ \boldsymbol{\mu} \in \mathbb{R}^p}} \left| \sum_{i \in G^1} (\boldsymbol{z}_i - \boldsymbol{\mu})(\boldsymbol{z}_i - \boldsymbol{\mu})^\top \right| > \delta_2 \tag{3.29}$$

Moreover, denoting $\lambda_1(\boldsymbol{A}) \geqslant \ldots \geqslant \lambda_p(\boldsymbol{A})$ the ordered eigenvalues of any p by p symmetric matrix $\boldsymbol{A}$ it is shown in [Seber, 2008, 10.56] that for $\boldsymbol{A} \succ 0$ and $\boldsymbol{B} \succeq 0$, $i = 1, \ldots, p$, it holds that

$$\lambda_i(\boldsymbol{A} + \boldsymbol{B}) \geqslant \lambda_i(\boldsymbol{A}). \tag{3.30}$$

Since $\sum_{i \in H^1 \setminus G^1} (\boldsymbol{z}_i^m - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))(\boldsymbol{z}_i^m - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))^\top \succeq 0$, it holds that

$$\begin{aligned} \left| (h-1)\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m) \right| &\geqslant \left| \sum_{i \in G^1} (\boldsymbol{z}_i^m - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))(\boldsymbol{z}_i^m - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))^\top \right| \\ &\geqslant \min_{\substack{G \subseteq \{1,\ldots,n\}, \\ \#G \geqslant p+1}} \left| \sum_{i \in G} (\boldsymbol{z}_i - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))(\boldsymbol{z}_i - \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))^\top \right| \\ &\geqslant \delta_2 > 0. \end{aligned} \tag{3.31}$$

Therefore, for a strictly positive constant δ_3 depending only on the entries of $\boldsymbol{Z}$ it holds that

$$0 < \delta_3 \leqslant |\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)|. \tag{3.32}$$

Furthermore, because each C-Step yields a new covariance matrix with smaller determinant than the initial covariance matrix [Rousseeuw and Van Driessen, 1999], we also have that

$$|\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)| \leqslant |\hat{\boldsymbol{\Sigma}}^{\mathrm{OGK}}(\boldsymbol{Z}^m)| \leqslant M_1. \tag{3.33}$$

It is shown in [Lopuhaä and Rousseeuw, 1991, Lemma 3.1] that whenever H^1 contains at least $p+1$ observations from $\boldsymbol{Z}$, one can always find a constant $M_3 > 0$ depending only on $\boldsymbol{Z}$ such that

$$||\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m)||_2 > M_3 \implies |\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)| > M_1,$$

But this would contradict (3.33), hence we conclude that

$$||\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m)||_2 \leqslant M_3 < \infty$$

Then, an argument similar to (3.29)–(3.31) establishes that, because the entries of $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$ and $\#G^1 \geqslant p+1$, $\lambda_p(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m))$ is bounded from below by a strictly positive constant δ_4 only depending on $\boldsymbol{Z}$. This, together with (3.33), also implies that $\lambda_1(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m))$ is bounded from above by a positive constant M_4 only depending on $\boldsymbol{Z}$. For any p by p symmetric matrix $\boldsymbol{A}$, denote $\mathrm{Tr}(\boldsymbol{A})$ the trace of $\boldsymbol{A}$. From the discussion above it follows that

$$\begin{aligned}
\sum_{i \in H^1} (\boldsymbol{z}_i^m)^\top \boldsymbol{z}_i^m &= (h-1)\,\mathrm{Tr}(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)) + h(\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))^\top \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m) \\
&\leqslant (h-1)p\lambda_1(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)) + h(\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m))^\top \hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m) \\
&\leqslant (h-1)pM_4 + hM_3. \qquad (3.34)
\end{aligned}$$

Therefore, there exist strictly positive constants M and M_0 depending only on $\boldsymbol{Z}$ such that

$$\sum_{i \in H^1} (y_i^m)^2 \leqslant M^2, \tag{3.35}$$

and

$$\sum_{i \in H^1} (\boldsymbol{x}_i^m)^\top \boldsymbol{x}_i^m \leqslant M_0^2. \tag{3.36}$$

For any index set H and matrix $\boldsymbol{Z}$, denote $\boldsymbol{Z}_H = \{\boldsymbol{z}_i\}_{i\in H}$. The estimates $\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)$ is given by the well known least square formula:

$$\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m) = \left((\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\right)^{-1} (\boldsymbol{X}^m_{H^1})^\top Y^m_{H^1},$$

where $\boldsymbol{X}$ includes a column of ones. Then, we have that

$$\begin{aligned}
\left|\left|\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)\right|\right|_2 &= \left|\left|\left((\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\right)^{-1} (\boldsymbol{X}^m_{H^1})^\top Y^m_{H^1}\right|\right|_2 \\
&\leqslant \lambda_1\left(\left((\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\right)^{-1}\right) \left|\left|(\boldsymbol{X}^m_{H^1})^\top Y^m_{H^1}\right|\right|_2 \\
&= \left(\lambda_p\left((\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\right)\right)^{-1} \left|\left|(\boldsymbol{X}^m_{H^1})^\top Y^m_{H^1}\right|\right|_2 . \qquad (3.37)
\end{aligned}$$

Denote the sub-components of $\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)$ as

$$\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m) = \begin{pmatrix} \hat{\boldsymbol{\Sigma}}^1(\boldsymbol{X}^m) & \hat{\boldsymbol{\Sigma}}^1(\boldsymbol{X}^m, Y^m) \\ (\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{X}^m, Y^m))^\top & \hat{\boldsymbol{\Sigma}}^1(Y^m) \end{pmatrix}$$

and write

$$\hat{\boldsymbol{\mu}}^1(\boldsymbol{Z}^m) = (\hat{\boldsymbol{\mu}}^1(\boldsymbol{X}^m), \hat{\boldsymbol{\mu}}^1(Y^m)).$$

Recall the identity

$$(\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1} = (h-1)\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{X}^m) + h\hat{\boldsymbol{\mu}}^1(\boldsymbol{X}^m)(\hat{\boldsymbol{\mu}}^1(\boldsymbol{X}^m))^\top. \qquad (3.38)$$

Since the smallest eigenvalue of any symmetric matrix $\boldsymbol{A}$ is smaller than the smallest eigenvalue of any principal sub-matrix of $\boldsymbol{A}$ ("Cauchy's interlace theorem") [Fisk, 2005], together with (3.30), we get from (3.38)

$$\begin{aligned}
\lambda_p\left((\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\right) &\geqslant (h-1)\lambda_p\left(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{X}^m)\right) \\
&\geqslant (h-1)\lambda_p\left(\hat{\boldsymbol{\Sigma}}^1(\boldsymbol{Z}^m)\right) \\
&\geqslant (h-1)\delta_4.
\end{aligned}$$

Combining the above equation with (3.35), (3.36) and (3.37) and using the Cauchy-Schwarz inequality yields

$$\begin{aligned} \left|\left|\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)\right|\right|_2 &\leqslant \frac{\left|\left|(\boldsymbol{X}^m_{H^1})^\top Y^m_{H^1}\right|\right|_2}{(h-1)\delta_4} \\ &\leqslant \frac{(M_0 M)^2}{(h-1)\delta_4}. \end{aligned} \tag{3.39}$$

Equation (3.39) shows that the initial regression estimator remains uniformly bounded.

Recall that

$$\begin{aligned} \sum_{i\in H^1} r_i^2(\boldsymbol{X}^m, Y^m, \hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)) &= \sum_{i\in H^1}(y_i^m)^2 - \\ &\quad (\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m))^\top (\boldsymbol{X}^m_{H^1})^\top \boldsymbol{X}^m_{H^1}\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m), \end{aligned}$$

so that

$$\sum_{i\in H^1} r_i^2(\boldsymbol{X}^m, Y^m, \hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)) \leqslant M^2.$$

The value of the (Det)LTS objective function evaluated at $\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)$ is

$$\begin{aligned} Q(\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)) &:= \sum_{i\leqslant h} r^2_{(i)}(\boldsymbol{X}^m, Y^m, \hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)) \\ &\leqslant \sum_{i\in H^1} r_i^2(\boldsymbol{X}^m, Y^m, \hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)) \\ &\leqslant M^2. \end{aligned}$$

Because the entries of $\boldsymbol{X}$ are in general position we have that

$$\inf_{\#G\geqslant p+1} \lambda_p(\boldsymbol{X}_G^\top \boldsymbol{X}) = \delta_5 > 0.$$

Furthermore, denote

$$\max_{\#G \geqslant p+1} ||\boldsymbol{X}_G^\top Y_G|| = N'.$$

Both δ_5 and N' only depend on $\boldsymbol{Z}$. Take $L > 0$ large enough such that

$$\inf_{t \geqslant L} (t^2 \delta_5 - N' t) \geqslant M^2 + 1.$$

Take now any $\boldsymbol{\beta}$ satisfying:

$$||\boldsymbol{\beta}|| > L.$$

Let H_β be the set of indexes corresponding to the h observations with smallest values of the squared residuals $\left((Y^m - \boldsymbol{X}^m\boldsymbol{\beta})\,(Y^m - \boldsymbol{X}^m\boldsymbol{\beta})^\top\right)_{ii}$ and set $G_{\boldsymbol{\beta}} = I^G \cap H_{\boldsymbol{\beta}}$. Then

$$\begin{aligned} Q(\boldsymbol{\beta}) &= \sum_{i \leqslant h} ((Y^m - \boldsymbol{X}^m\boldsymbol{\beta})(Y^m - \boldsymbol{X}^m\boldsymbol{\beta})^\top)_{(ii)} \\ &\geqslant \sum_{i \in G_{\boldsymbol{\beta}}} ((Y^m - \boldsymbol{X}^m\boldsymbol{\beta})(Y^m - \boldsymbol{X}^m\boldsymbol{\beta})^\top)_{(ii)} \\ &= Y_{G_{\boldsymbol{\beta}}}^\top Y_{G_{\boldsymbol{\beta}}} + \boldsymbol{\beta}^\top (\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top \boldsymbol{X}_{G_{\boldsymbol{\beta}}})\boldsymbol{\beta} - 2\boldsymbol{\beta}^\top (\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top Y_{G_{\boldsymbol{\beta}}}). \end{aligned} \tag{3.40}$$

Since $\boldsymbol{\beta}^\top (\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top \boldsymbol{X}_{G_{\boldsymbol{\beta}}}) \geqslant ||\boldsymbol{\beta}||_2 \lambda_p \left(\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top \boldsymbol{X}_{G_{\boldsymbol{\beta}}}\right) \geqslant ||\boldsymbol{\beta}||_2 \delta_5$ and Cauchy-Schwarz yields $\boldsymbol{\beta}^\top (\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top Y_{G_{\boldsymbol{\beta}}}) \leqslant ||\boldsymbol{\beta}||_2 \left|\left|\boldsymbol{X}_{G_{\boldsymbol{\beta}}}^\top Y_{G_{\boldsymbol{\beta}}}\right|\right| \leqslant ||\boldsymbol{\beta}||_2 N'$, it follows that

$$Q(\boldsymbol{\beta}) \geqslant ||\boldsymbol{\beta}||_2\, \delta_5 - ||\boldsymbol{\beta}||\, N' > M^2 \geqslant Q(\hat{\boldsymbol{\beta}}^{\text{init}}(\boldsymbol{Z}^m)). \tag{3.41}$$

We conclude that:

$$\left|\left|\hat{\boldsymbol{\beta}}^{\text{RDetLTS}}(\boldsymbol{Z}^m)\right|\right|_2 \leqslant L. \tag{3.42}$$

The exposition above shows that the finite sample breakdown point of $\hat{\boldsymbol{\beta}}^{\text{RDetLTS}}(\boldsymbol{Z}^m)$ is at least $(n-h+1)/n$ when the entries of $\boldsymbol{Z}$ are in general position in $\mathbb{R}^p$. This also holds for $\hat{\boldsymbol{\beta}}^{\text{DetLTS}}(\boldsymbol{Z}^m)$ since one step re-weighting

preserves the finite sample breakdown point of the initial estimates [Lopuhaä and Rousseeuw, 1991]. □

	DetLTS	FastLTS
(Intercept)	-5.2E+01	-5.2E+01
General Health Condition (Cumulative percentage)	-6.1E-02	-6.1E-02
Times Ate Cake/pie/cookies Last Month	-1.6E-01	-1.6E-01
Times Ate Ice Cream/frozen Desserts Last Month	-3.5E-02	-3.5E-02
Times Walked at Least 10 Min for Leisure past 7 Days	-7.9E-02	-7.2E-02
Average Length of Time Walked for Leisure	-3.8E-03	-3.4E-03
Times Ate Fruit in past Month	-8.4E-02	-8.4E-02
Days Moderate Physical Activity in past Week	-3.4E-01	-3.2E-01
Time per Day of Moderate Physical Activity	2.4E-03	2.0E-03
Times Ate French Fries, Ham Fries, Hash Browns in Past Month	5.8E-02	5.8E-02
Times Ate Vegetables in past Month	-7.4E-02	-7.5E-02
Times Saw Md in past Year	4.3E-02	7.2E-03
Usual Hours Worked per Week	2.0E-02	2.0E-02
Time Working at Main Job (Months)	-1.0E-04	-5.2E-05
Serious Psychological Distress (Kessler scale)	-2.1E-01	-1.9E-01
Educational Attainment (Cumulative percentage)	4.0E-02	3.9E-02
Time Lived at Current Address (Months)	7.2E-04	5.9E-04
Respondent's Earnings Last Month	-8.8E-05	-8.7E-05
Household's Total Annual Income	9.3E-06	9.5E-06
Height	5.6E+01	5.6E+01
Age	-2.4E-02	-1.9E-02
Weight	5.1E-01	5.0E-01
Unadjusted Daily Teaspoons of Added Sugar in Pastries	1.2E+00	1.2E+00
Unadjusted Daily Teaspoons of Added Sugar in Beverages	-5.4E-02	-4.8E-02
Daily Cup Equivalents of Fruits and Vegetables Excluding Beans	3.2E+00	3.2E+00
Daily Teaspoons of Added Sugar	4.2E-02	3.8E-02

Table 3.6: Table of fitted coefficients for DetLTS (first column) and FastLTS (second column). The last column depicts the FastLTS standard errors.

Chapter 4

Multivariate Functional Halfspace Depth

4.1 Introduction

Nowadays, functional data are frequently observed and many statistical methods have been developed to retrieve useful information from these data sets. Typically, the observed data consist of a set of N curves, each measured at different time points $t_1, \ldots, t_T$. For an overview, see [Ramsay and Silverman, 2005, Ferraty and Vieu, 2006]. Basic questions of interest in functional data analysis (FDA) are (i) the estimation of the central tendency of the curves, (ii) the estimation of the variability among the curves, (iii) the detection of outlying curves, as well as (iv) classification and clustering of such curves.

In this chapter we consider *multivariate functional data*. We observe for all observation units at each time point a K-dimensional vector of measurements, which arise from an underlying set of K curves. A popular example is the bivariate gait data set, which contains the simultaneous variation of the hip and knee angles for 39 children at 20 equally space time points [Ramsay and Silverman, 2005]. [Berrendero et al., 2011] have $K = 3$ when recording daily temperature functions at 3, 9 and 12cm below the surface during $N = 21$ days. [Sangalli et al., 2009] and [Pigoli and Sangalli, 2012] present several multivariate functional data from medical studies. Bivariate U.K. weather data are studied in Section 4.3.2.

Different types of multivariate functional data arise by computing additional

curves, starting from one observed set of univariate functional data. A well-studied situation is the addition of the first order *derivatives* which provides additional information on the shape of the curves and consequently is interesting to detect curves with an outlying shape [Cuevas et al., 2007]. Note that this is different from a common practice in chemometrics, where observed spectral data are often *replaced* by their first-order derivatives in order to eliminate baseline features. Also higher order derivatives could be added. This has been applied in the Berkeley growth data set [Ramsay and Silverman, 2005], which contains the heights of children and the estimated acceleration curves that correspond to the second-order derivatives.

In this chapter we introduce to the depth calculation the inclusion of other functions of the original set of curves (such as warping functions, derivatives, integrals,...) which allows us to obtain more powerful conclusions about the data-driven process. Some interesting functions are obtained from a *warping* procedure, which often precedes the analysis of functional data. Typically some warping method (also known as curve alignment) is applied to the observed curves as a preprocessing step, but no further information is retained from this analysis. In [Slaets et al., 2012] it is shown how the information from the warping procedure can be incorporated into a clustering method for functional data. In Section 4.4.3 we show the benefits of a multivariate analysis of the warped data together with the curves obtained via the warping function.

A different augmentation of the data is presented in Section 4.3. It contains the analysis of a real data set which consists of acceleration signals over time from an industrial machine [De Ketelaere et al., 2011]. Most of the observed curves, see Figure 4.1(a), follow a similar nonlinear pattern, but we also notice several curves with a deviating trend, most prominently at the final stage of the production. In addition to these acceleration signals, we do not use their derivatives but rather the *integrated* curves as they represent the underlying velocity, see Figure 4.1(b). Also here, we observe a global structure as well as deviating signals. On both plots we have added the cross-sectional mean curve (dashed line). Further we have plotted our new estimator for the central tendency of the curves, shown as a solid dark line. It is already obvious that these estimates are less influenced by the outlying curves. For the velocity curves, the effect is less pronounced as the outlying curves occur in both directions of the central pattern.

Our approach to estimate the central tendency of multivariate functional data is based on the concept of depth. Depth functions were initially defined for multivariate data. They provide an ordering from the center outwards such that the most central object gets the highest depth value and the least central objects the smallest depth. More recently, several notions of depth have been proposed for univariate functional data, such as the Fraiman and Muniz depth (FM)

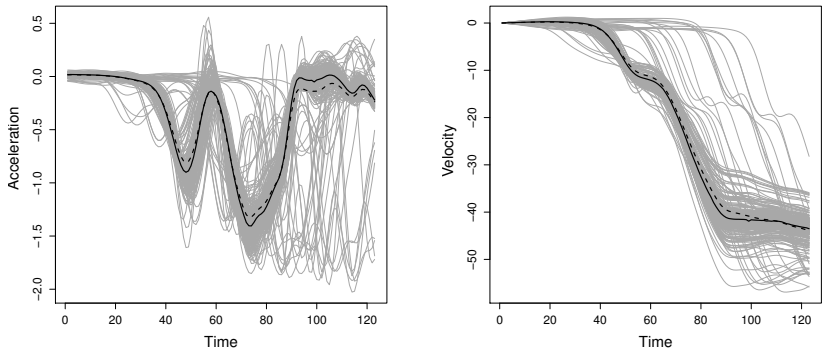


Figure 4.1: Acceleration (left) and velocity signals (right), with cross-sectional mean curve (dashed line) and depth-based median curve (solid dark line).

[Fraiman and Muniz, 2001], the h-mode and random projection depth (RP) [Cuevas et al., 2007], the band depth and modified band depth (MBD) [López-Pintado and Romo, 2009] and the half-region depth [Lopez-Pintado and Romo, 2011]. The FM depth and MBD depth are quite similar, as they both consider a (univariate) depth function at each time point t and define the functional depth as the *average* of these depth values over all time points. [Cuevas et al., 2007] have proposed to consider the curves and their derivatives, yielding the bivariate random projection depth (RPD). For a number of random projections, they project both sets of curves on each direction, apply a multivariate depth function on the bivariate sample and finally average the depth values over the random projections.

We generalize several of these ideas by constructing a depth function for K-variate samples of curves, which we define as the multivariate functional depth (MFD). Our definition averages a multivariate depth function over the time points, but in addition it includes a weight function. This weight function can be chosen as to account for variability in amplitude, to adapt to the functional nature of the data. More specifically we choose Tukey's halfspace depth [Tukey, 1975] as the building block, which leads to the multivariate functional halfspace depth (MFHD).

The rest of this chapter unfolds as follows. In Section 4.2 we give some general definitions and proprieties of MFHD. Next, in Section 4.3 we illustrate the use of MFHD on two data examples. In Section 4.4 we use numerical simulations to compare the performance of MFHD to some methods designed for functional data that are popular in the growing literature on the subject. Finally, Section 4.5 offers some closing remarks.

4.2 Definition and properties of multivariate functional depth

4.2.1 Notation

Consider a K-variate (finite K), real-valued stochastic process of continuous functions $\boldsymbol{\mathcal{Y}} = (\mathcal{Y}_1, \ldots, \mathcal{Y}_K)$ with for $j = 1, \ldots, K$, $\mathcal{Y}_j : U \to \mathbb{R} : t \mapsto \mathcal{Y}_j(t)$ continuous on a compact interval U and denote its cumulative distribution by $F_{\boldsymbol{\mathcal{Y}}}$. Thus, for every finite set of time points $t_1, \ldots, t_T \in U$, $(\boldsymbol{\mathcal{Y}}(t_1), \ldots, \boldsymbol{\mathcal{Y}}(t_T))$ is a random variable on $\left(\mathbb{R}^K\right)^T$ and at each time point $t \in U$, $\boldsymbol{\mathcal{Y}}(t)$ is a K-variate random variable with associated cumulative distribution function (cdf) $F_{\boldsymbol{\mathcal{Y}}(t)}$.

Real numbers, vectors, continuous functions on an interval U and vectors of functions are all used in conjunction with each other. To avoid confusion, we provide an overview of the notation that is used throughout this chapter. The set of continuous functions on U is denoted by $\mathcal{C}(U)$. Elements thereof and their graphs are denoted by capital letters (e.g. X). For K-vectors of continuous functions in $\mathcal{C}(U)^K$ and their graphs, bold capital letters are used (e.g. $\boldsymbol{X}$) or the vector notation $(X_1, \ldots, X_K)$ where $X_i \in \mathcal{C}(U)$. The function value of a curve X at a time point t is denoted by $X(t) \in \mathbb{R}$. The vector of function values of an element $\boldsymbol{X}$ in $\mathcal{C}(U)^K$ at a time point t is denoted by $\boldsymbol{X}(t) = (X_1(t), \ldots, X_K(t))$. The empirical cumulative distribution function based on a sample $\{\boldsymbol{Y}_1(t), \ldots, \boldsymbol{Y}_N(t)\}$ each with the same distribution as $\boldsymbol{\mathcal{Y}}(t)$ is denoted by $F_{\boldsymbol{\mathcal{Y}}(t),N}$. For vectors in $\mathbb{R}^K$, bold lowercase letters are used (e.g. $\boldsymbol{a}$) or the vector notation $(a_1, \ldots, a_K) \in \mathbb{R}^K$. For matrices capital letters early in the alphabet are used (e.g. $\boldsymbol{A}$), while later letters (e.g. $\boldsymbol{X}$) are reserved for curves. For real numbers, lowercase letters are used (e.g. a).

4.2.2 General Definition

A depth function provides an ordering from the center outwards such that the most central objects get the highest depth and the least central objects the smallest depth. Let $D(\cdot; F_{\boldsymbol{\mathcal{X}}}) : \mathbb{R}^K \to [0, 1]$ be a statistical depth function for the probability distribution of a K-variate random vector $\boldsymbol{\mathcal{X}}$ with cdf $F_{\boldsymbol{\mathcal{X}}}$, according to [Zuo and Serfling, 2000]. Associated with the depth function is the depth region $D_\alpha(F_{\boldsymbol{\mathcal{X}}})$ at level $\alpha \geqslant 0$, defined as $D_\alpha(F_{\boldsymbol{\mathcal{X}}}) = \{\boldsymbol{x} \in \mathbb{R}^K : D(\boldsymbol{x}; F_{\boldsymbol{\mathcal{X}}}) \geqslant \alpha\}$.

The multivariate *functional* depth combines the local depths of $\boldsymbol{\mathcal{Y}}(t)$ at each time point $t \in U$ and includes a weight function that may be specified specifically according to the purposes of the analysis. It will now be necessary to justify the definition of the finite sample version of the multivariate depth function we

will adopt in Definition 2 below by briefly recalling some key properties of its population counterpart and defined in greater details in [Claeskens et al., 2014].

Definition 1. *Consider a K-variate stochastic process $\{\boldsymbol{\mathcal{Y}}(t), t \in U\}$ on $\mathbb{R}^K$ with cdf $F_{\boldsymbol{\mathcal{Y}}}$ that generates continuous paths in $\mathcal{C}(U)^K$. Let D be a statistical depth function on $\mathbb{R}^K$ and w a weight function that is defined on U and integrates to one, this weight function may or may not depend on $F_{\boldsymbol{\mathcal{Y}}(t)}$, $t \in U$. Take an arbitrary $\boldsymbol{X} \in \mathcal{C}(U)^K$. The multivariate functional depth (MFD) of $\boldsymbol{X}$ is defined as*

$$MFD(\boldsymbol{X}; F_{\boldsymbol{\mathcal{Y}}}) \quad = \quad \int_U D(\boldsymbol{X}(t); F_{\boldsymbol{\mathcal{Y}}(t)}) \cdot w(t)\, dt. \tag{4.1}$$

A first example for the weight is a constant times an indicator for a range of interest. This allows for example to eliminate the effect of a start-up phase in an industrial process, or to remove imprecise measurements during certain regions which often happens for spectral data. A second example takes the local changes in the amount of variability in amplitude (vertical variability) into account by defining

$$w(t) = w_\alpha(t; F_{\boldsymbol{\mathcal{Y}}(t)}) = \mathrm{vol}\{D_\alpha(F_{\boldsymbol{\mathcal{Y}}(t)})\} \,/\, \int_U \mathrm{vol}\{D_\alpha(F_{\boldsymbol{\mathcal{Y}}(u)})\} du, \tag{4.2}$$

which is proportional to the volume of the depth region at time point t. This implies that for regions where all curves nearly coincide the weight is small, heuristically, the order of the curves does not matter much here. For regions where the amplitude variability is large, there is a visual ordering of the curves, and the influence of those regions on the functional depth will be large.

When the weight function (4.2) is considered, we denote the corresponding depth by MFD(α). Note that for many cases the definition of MFD with this choice of weight does not depend on α. In general, the value of α is irrelevant when at each time point t the volumes of the depth regions are proportional to a fixed function of α. In particular, for most depth functions, it holds that at unimodal elliptic symmetric distributions, the contours of the depth regions coincide with density contours, which implies that the choice of α in MFD(α) becomes irrelevant (at least at the population level) when this distribution remains the same up to a scaling constant.

In Theorem 1 from [Claeskens et al., 2014], the authors show that the multivariate functional depth satisfies some key properties, adapted to a functional data context, that were put forward by [Zuo and Serfling, 2000]:

Theorem 1. *Assume that the depth function D satisfies the four properties listed in [Zuo and Serfling, 2000], i.e. affine invariant, maximal at the center, monotone relative to the deepest point and vanishing at infinity. Then MFD, as defined in Definition 1, is a statistical depth function satisfying the following key properties:*
(i) Affine invariance (invariance w.r.t. the underlying coordinate system). If the weight function w is affine invariant, then

$$MFD(\boldsymbol{X}; F_{\boldsymbol{\mathcal{Y}}}) = MFD(\boldsymbol{A}\boldsymbol{X}_{(ct+d)} + \tilde{\boldsymbol{X}}_{(ct+d)}; F_{\boldsymbol{A}\boldsymbol{\mathcal{Y}}_{(ct+d)}+\tilde{\boldsymbol{X}}_{(ct+d)}}),$$

with $U = [l, u]$, $\boldsymbol{A}\boldsymbol{\mathcal{Y}}_{(ct+d)} + \tilde{\boldsymbol{X}}_{(ct+d)}$ the stochastic process $\{\boldsymbol{A}\boldsymbol{\mathcal{Y}}(\frac{s-d}{c}) + \tilde{\boldsymbol{X}}(\frac{s-d}{c}), s \in S = [cl + d, cu + d]\}$ for any constants $c \in \mathbb{R}_0$, $d \in \mathbb{R}$, any vector of functions $\tilde{\boldsymbol{X}} \in \mathcal{C}(U)^K$ and any matrix $\boldsymbol{A} \in \mathbb{R}^{K\times K}$ with $\det(\boldsymbol{A}) \neq 0$, and $\tilde{\boldsymbol{X}}_{(ct+d)}$ the curve $\{s, \tilde{\boldsymbol{X}}(\frac{s-d}{c})\}$, with $s \in S = [cl + d, cu + d]$.
(ii) Maximality at the center. For a uniquely defined $\boldsymbol{\Theta} \in C(U)^K$ such that $\boldsymbol{\Theta}(t)$ is a symmetry point in which D is maximal at every $t \in U$, it holds that $MFD(\boldsymbol{\Theta}; F_{\boldsymbol{\mathcal{Y}}}) = \sup_{\boldsymbol{X}\in\mathcal{C}(U)^K} MFD(\boldsymbol{X}; F_{\boldsymbol{\mathcal{Y}}})$.
(iii) Monotonicity relative to the deepest point. Let $\boldsymbol{\Theta} \in C(U)^K$ such that $\boldsymbol{\Theta}(t)$ is a deepest point at every $t \in U$, then for any $a \in [0, 1]$, $MFD(\boldsymbol{X}; F_{\boldsymbol{\mathcal{Y}}}) \leqslant MFD(\boldsymbol{\Theta} + a(\boldsymbol{X} - \boldsymbol{\Theta}); F_{\boldsymbol{\mathcal{Y}}})$.
(iv) Vanishing at infinity. For $1 \leqslant k \leqslant K$ and for a series of curves $X_{n,k}$ with $\lim_{n\to\infty} |X_{n,k}(t)| = \infty$ for almost all time points t in U: $\lim_{n\to\infty} MFD(X_{n,k}; F_{\boldsymbol{\mathcal{Y}}}) = 0$.

The affine invariance holds for all specified weight functions we consider below. In general, when the weight is invariant with respect to transformations $t \mapsto \boldsymbol{A}(t)\boldsymbol{X}(t) + \tilde{\boldsymbol{X}}(t)$, the same holds for MFD. In the original multivariate setting, the fourth property, 'vanishing at infinity', requires that for a vector $\boldsymbol{x} \in \mathbb{R}^K$ the depth of $\boldsymbol{x}$ should converge to 0 for $\|\boldsymbol{x}\| \to \infty$. When a curve behaves in accordance with the sample on the majority of the interval and, e.g., converges to infinity near the border, one might not wish to attribute zero depth. The vanishing at infinity property for functional depth holds as stated in (iv).

4.2.3 Finite sample definition

A general multivariate depth as a building block

In practice one does not observe curves, but rather curve evaluations at a set of time points $t_1 < t_2 < \ldots < t_T$ in $U = [t_1, t_T]$, not necessarily equidistant.

Definition 2. *For a sample of multivariate curve observations* $\{\boldsymbol{Y}_1(t_j), \ldots, \boldsymbol{Y}_N(t_j); j = 1, \ldots, T\}$*, with at each time point* t *cdf* $F_{\boldsymbol{\mathcal{Y}}(t),N}$*, the sample multivariate functional depth at* $\boldsymbol{X} \in \mathcal{C}(U)^K$ *is defined by, with* $t_0 = t_1$*,* $t_{T+1} = t_T$ *and* $W_j = \int_{(t_{j-1}+t_j)/2}^{(t_j+t_{j+1})/2} w(t)dt$*,*

$$MFD_N(\boldsymbol{X}) = \sum_{j=1}^{T} D(\boldsymbol{X}(t_j); F_{\boldsymbol{\mathcal{Y}}(t_j),N}) W_j. \tag{4.3}$$

Special cases. For a constant weight $w(t) = w$ for all $t \in U$, $W_j = w \cdot (t_{j+1} - t_{j-1})/2$. For a non constant and affine invariant depth function, one can use

$$\begin{aligned} W_j \quad = \quad & \mathrm{vol}\{D_\alpha(F_{\boldsymbol{\mathcal{Y}}(t_j),N})\}(t_{j+1} - t_{j-1}) \\ & / \Big\{ \sum_{j=1}^{T} \mathrm{vol}\{D_\alpha(F_{\boldsymbol{\mathcal{Y}}(t_j),N})\}(t_{j+1} - t_{j-1}) \Big\}. \end{aligned}$$

Based on MFD_N we can estimate the global pattern of the observed curves by means of the $\boldsymbol{\Theta}(t) \in C(U)^K$ which attains maximal MFD_N. For a general depth function D it might however be not straightforward to compute this median curve. In that case one can approximate $\boldsymbol{\Theta}(t)$ by the curve with maximal MFD_N among all observed curves. Apart from estimating the global pattern of the curves, we are often interested in the variability of the curves. Our depth-based approach allows to visualize this dispersion by means of the *central regions*, introduced in [López-Pintado and Romo, 2009]. The β-central region consists of the band delimited by the $[N\beta]$ curves with highest depth. If we draw the 25%, 50% and 75% central regions, we obtain a representation of the data as in the enhanced functional boxplot of [Sun and Genton, 2011, Sun and Genton, 2012]. See Section 4.3 for examples. Based on these central regions, we define for each univariate set of curves their *dispersion curves* $s_\beta(t)$ as the width of the β-central region at each t. Note that the dispersion curves are defined on each of the univariate curves, but the underlying computation of the central regions is based on the MFD. The $t \mapsto s_{0.5}(t)$ dispersion curve can be considered as a kind of functional IQR, as explained in [Sun and Genton, 2011]. A related concept, the scale curve, is defined in [López-Pintado et al., 2010]. It measures the area of the central region for β ranging from 0 to 1, and could be considered here as well. The β-trimmed mean and the β-trimmed variance (the mean and variance of all curves in the β-central region), see [**?**], can be extended in a straightforward way too.

Halfspace depth as a building block

We now define the finite-sample multivariate functional halfspace depth (MFHD_N) as in Definition 2 with D the sample halfspace depth based on $\{\boldsymbol{Y}_1(t), \ldots, \boldsymbol{Y}_N(t)\}$ [Tukey, 1975],

$$\text{HD}(\boldsymbol{X}(t); F_{\boldsymbol{Y}(t),N}) = \frac{1}{N} \min_{\boldsymbol{u}\in\mathbb{R}^K, \|\boldsymbol{u}\|=1} \#\{\boldsymbol{Y}_n(t), n = 1, \ldots, N : \boldsymbol{u}^\top \boldsymbol{Y}_n(t) \geqslant \boldsymbol{u}^\top \boldsymbol{X}(t)\}.$$

The finite-sample Tukey median is defined as the center of gravity of the deepest depth region. The median curve of the sample $\{\boldsymbol{Y}_1(t_j), \ldots, \boldsymbol{Y}_N(t_j); j = 1, \ldots, T\}$ is defined as the Tukey median at each time point.

In MFHD, we choose the halfspace depth [Tukey, 1975] as depth function because it satisfies the requirements of a building block for the functional depth as stated in Theorem 1. Consequently MFHD is affine invariant, maximal at the point (curve) of symmetry, monotone relative to the deepest point, and vanishing at infinity. An additional advantage of HD is its robustness with respect to outliers. The influence function of the halfspace depth of any multivariate point in $\mathbb{R}^K$ is bounded [Romanazzi, 2001] and the deepest point (Tukey median) has a positive breakdown value between $1/(K+1)$ and $1/3$ at absolutely continuous distributions [Chen and Tyler, 2002]. Finally, fast algorithms exist for the computation of HD at multivariate data, as well as for the depth regions and for the Tukey median.

Exact computation of the MFHD_N can be done with fast algorithms for the halfspace depth up to dimension $K = 4$ [Bremner et al., 2008]. To compute the weight function (4.2), the algorithms developed in [Hallin et al., 2010] allow the computation of the depth contours up to dimension at least $K = 5$. In this chapter we used the R-packages `depth` and `aplpack` which implement fast algorithms for bivariate and trivariate data [Rousseeuw and Ruts, 1996, Rousseeuw and Ruts, 1998, Rousseeuw and Struyf, 1998, Rousseeuw et al., 1999a]. Approximate halfspace depth in higher dimensions can be computed by means of the random Tukey depth [Cuesta-Albertos and Nieto-Reyes, 2008], but it is no longer affine invariant.

A β trimmed mean of curves is defined by first assigning a depth value to each curve and by omitting the $[N\beta]$ of curves with the lowest depth. A cross-sectional average of the remaining curves may then be computed. The deepest curve, also called the median, is defined as the curve with maximal MFHD. It can easily be shown that it corresponds to the curve Θ for which at each time point t, $\Theta(t)$ is a deepest value, obtained by the center of mass of the set of values with maximum halfspace depth at time t. Under some assumptions, the population median can be shown to exist and to be continuous.

To decide upon the value of α in (4.2), one could consider some empirical quantile of the HD values at each time point, and take, e.g., their minimum or average. For example, if we set α equal to the minimal median of the HD values, the resulting depth contours cover at least half of the data at each time point. Another possibility is to rely on the probability coverage of the depth contours, which can be computed exactly at some multivariate distributions. At bivariate normal data, it can be derived that $\alpha = 1/8$ gives a coverage of 48% [Rousseeuw and Ruts, 1999]. For univariate normal data, a 50% coverage is attained at $\alpha = 1/4$. We have used these as default values in our data examples and we have verified that the coverage was indeed around 50% at all time points.

4.3 Data examples

4.3.1 Industrial data

Central curve estimation

We illustrate our new depth function on an industrial data set that produces one part during each cycle [De Ketelaere et al., 2011]. The behavior of the cycle as monitored by an accelerometer provides a fingerprint of the cycle and, related, of the quality of the produced part. If a deviating acceleration signal occurs, the process owner should be warned. Figure 4.2 shows the acceleration signal of $N = 224$ parts measured during 120ms (in gray). Measurements are available every millisecond, hence the time signal ranges from $t_1 = 1$ up to $t_T = 120$. On this plot we see several curves with a deviation pattern, most prominently at the final stage of the production.

To estimate the central pattern of the data, we first computed the mean curve, displayed in green on Figure 4.2. It gives a quite good representation of the main features of the curves, but it is clearly attracted by the outlying values during the last 30ms of the cycle. Next, we computed the MFHD on these original set of curves, with $\alpha = 0.25$. As we only have univariate measurements at each time point, this boils down for each curve to compute its univariate halfspace depth at each time point and to take the weighted average of these depth values. The curve which attains the maximal MFHD is depicted in Figure 4.2 in dark red. We see that this deepest curve is not attracted by the outlying values at the end of the cycle. Also the estimates in the valleys around time points 50 and 75 are lower than those of the mean curve, illustrating the robustness of the deepest curve towards the upward contamination values in these regions. Finally we also consider the 25% trimmed mean curve, obtained by trimming the 25% curves with lowest depth and taking the pointwise mean

of the remaining curves (displayed in orange). This trimmed mean is hardly distinguishable from the deepest curve.

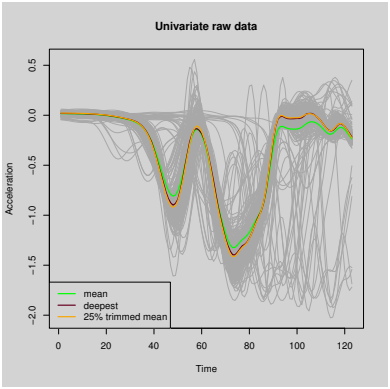
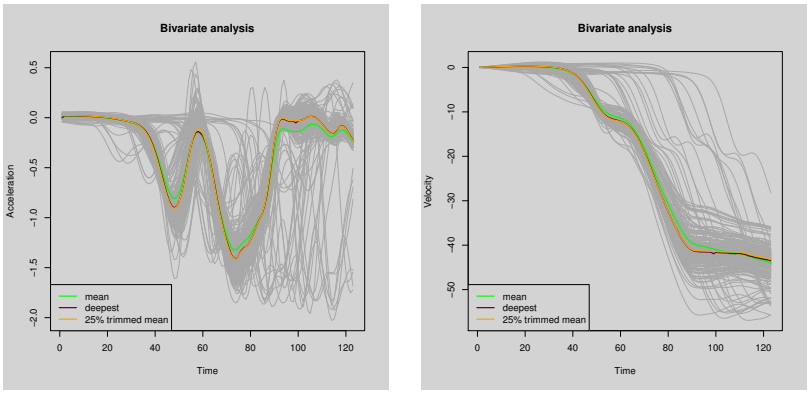


Figure 4.2: Mean curve, deepest curve and 25% trimmed mean based on the univariate MFHD.

Next, we performed a bivariate analysis on this data set. We could consider the derivatives of the curves as additional information, but in this example, we decided to use the integrated curves instead. As the velocity at time t_j, $V(t_j) = \int_{-\infty}^{t_j} A(t)dt$ with $A(t)$ the acceleration at time t, we approximated the velocity by $V(t_j) \approx V(t_{j-1}) + (A(t_{j-1}) + A(t_j))/2$ starting with $V(t_1) = 0$. Note that the choice of the integration constant is not important here, due to the affine invariance of MFHD. The resulting velocity curves can be seen in Figure 4.3(b). Also here we see several curves whose velocity is unusual during a large part of the cycle. The mean curve is slightly affected by these outliers. Computing the MFHD on the bivariate data $(A(t), V(t))$ yields a deepest set of curves, again printed in dark red on Figure 4.3 for the acceleration and velocity curves. There are no huge differences between the deepest curve in Figure 4.2, but it lies closer to the (more efficient) trimmed mean.

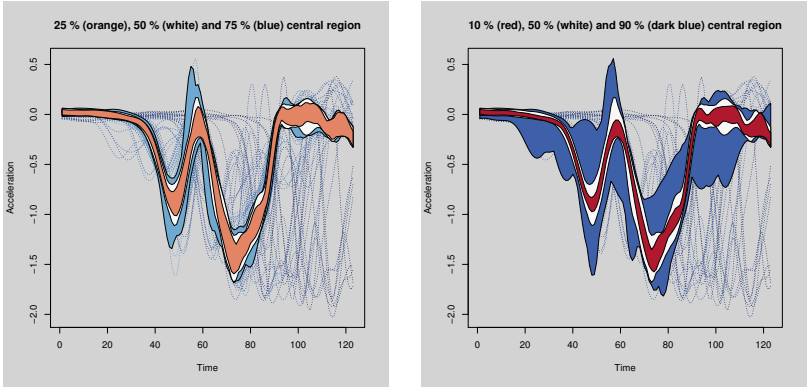
Apart from estimating the global pattern of the curves, we are also interested in the variability of the curves. The β-central region consists of the band delimited by the β curves with highest depth. If we draw the 25%, 50% and 75% central regions, we obtain a representation of the data as in the enhanced functional boxplot of [Sun and Genton, 2011]. As before, we can construct these regions based on the original curves $A(t)$ only, which yields Figure 4.4(a). Similarly the 10%, 50% and 90% central regions are depicted in Figure 4.4(b). It is obvious that the 90% central region contains outlying curves and hence increases the volume of that central region. Based on the bivariate analysis, we obtain Figures



(a) (b)

Figure 4.3: Mean curve, deepest curve and 25% trimmed mean based on the bivariate MFHD.

4.5(a)-(b) for the acceleration curves and (c)-(d) for the velocity curves. Now, we see a more important difference between the univariate and the bivariate analysis, as the 50% and 75% central regions of the acceleration curves are quite different between 40ms and 60ms.



(a) (b)

Figure 4.4: Central regions for the acceleration curves based on the univariate analysis.

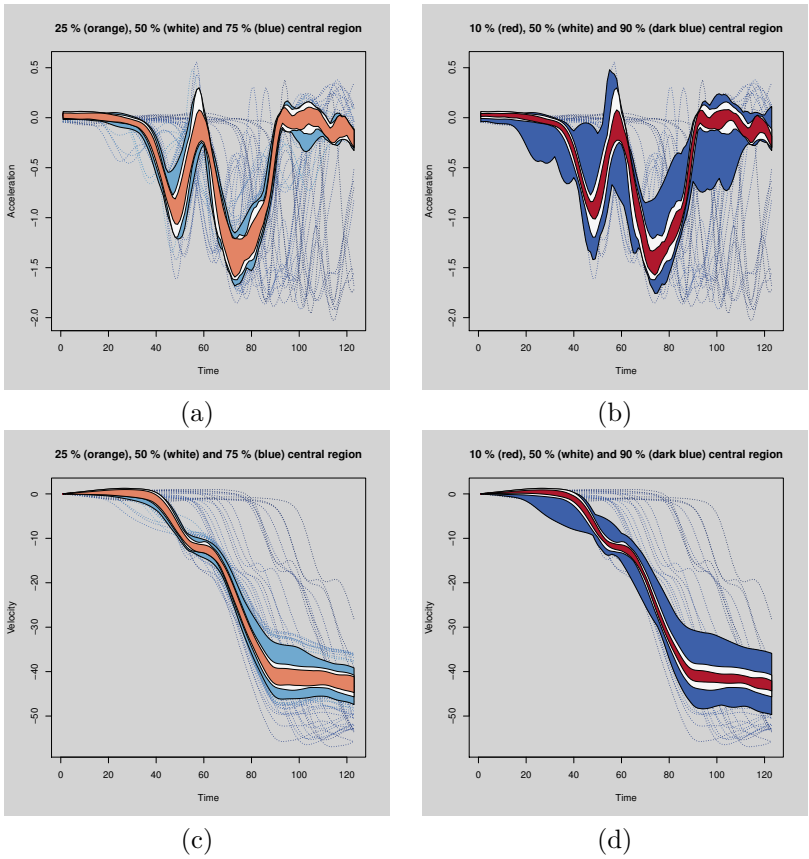


Figure 4.5: Central regions for (a - b) the acceleration curves based on the bivariate analysis; and (c - d) central regions for the velocity curves.

To understand the difference between the univariate and the bivariate analysis, we first compare the univariate and bivariate MFHD values for all curves, shown in Figure 4.8. We see a global monotone trend showing that curves with a low univariate MFHD depth also have a low bivariate MFHD depth, but the relation is certainly not strictly monotone.

Let us focus on two specific curves, with labels 112 and 207, indicated in Figure 4.7, for comparison together with the deepest curve. Curve 207 clearly has a completely different acceleration and velocity pattern than the trend observed on the regular curves. Only in the beginning of the process, the acceleration and velocity are small and comparable with the others. Not surprisingly, both

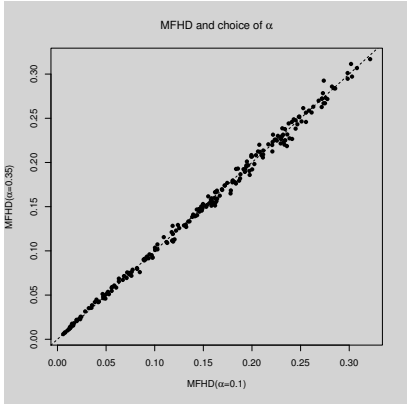


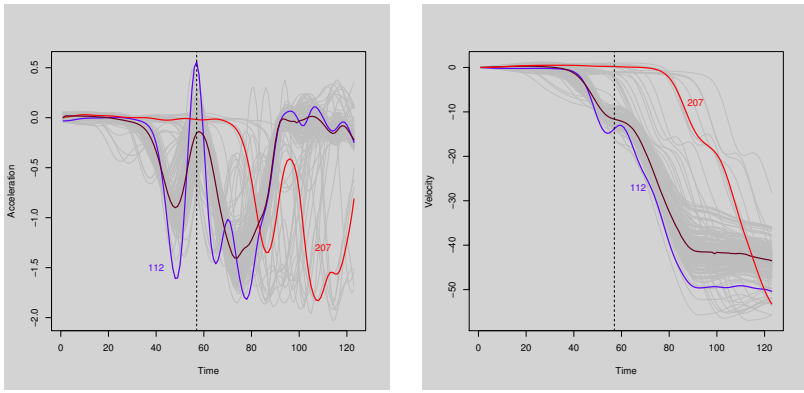
Figure 4.6: Influence of α on the MFHD values.

the univariate and bivariate MFHD are small for this curve, as we observe in Figure 4.8. Curve 112 shows a different, deviating pattern. From Figure 4.7(a) we notice that it attains larger acceleration values at the peaks around 47ms and 57ms, and one additional oscillation between 60ms and 80ms. Consequently its univariate depth, only based on this information, is somewhat lower but it is not extremely small. To be more precise, the univariate MFHD of curve 112 has rank 45 (out of 224). When we include the information given by the velocity curves, we see from Figure 4.7(b) that the velocity of curve 112 is outlying on almost the whole time domain. This yields a bivariate MFHD with rank 15, which is close to the rank of the bivariate MHFD of curve 207 which equals 12. As a result, the 75% central region based on the bivariate depth does not include the curves with large peaks around 47ms and 57ms, whereas the univariate-based central region does include them.

Note that our definition of MFHD depends on the choice of the level of the depth contours α in the weighting function (4.4). However, we noticed that our analysis is usually not very sensitive to this choice, as long as α is not taken too small such that outlying curves are not included in the depth contour. Figure 4.6 shows a scatterplot of the MFHD depth with $\alpha = 0.35$ versus the depth values for $\alpha = 0.1$. We see that they are very similar.

Outlier detection

To detect outlying curves, we can follow two strategies. First it can be argued that the curves with lowest MFHD are potential outliers. This is for example the



(a) (b)

Figure 4.7: (a) Acceleration and (b) velocity curves with two outlying curves and the deepest curve.

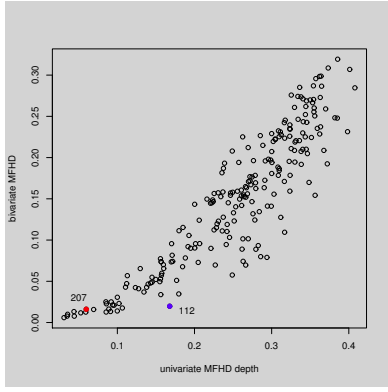
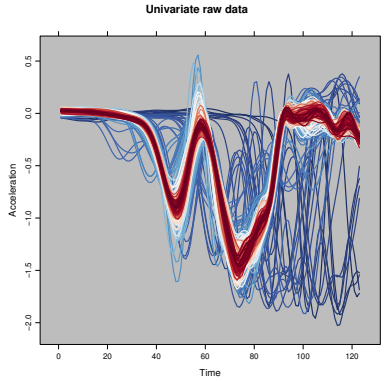


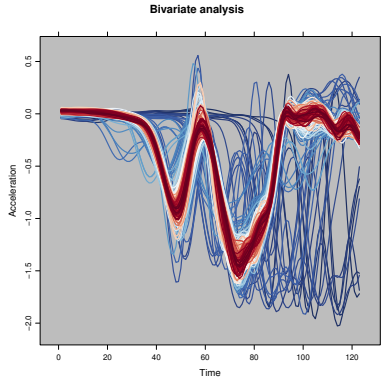
Figure 4.8: Univariate versus bivariate MFHD.

approach considered in [Febrero et al., 2008]. As depth provides an ordering of the curves from the center outwards, we indeed expect that outlying curves have a low depth. This was also empiricially verified in Section 4.3.1. To visualise these potential outliers, we color all the curves according to their depth, yielding a so-called rainbow plot [Hyndman and Shang, 2010]. We first order the curves from maximal to minimal depth. Then we go from dark red for the deepest curve, to white for the curve with rank $N/2$, and move to dark blue for the curve with minimal depth. This yields Figure 4.9(a) based on the univariate

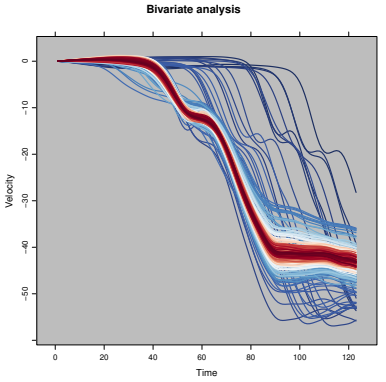
MFHD, and Figure 4.9(b) and (c) based on the bivariate MFHD. We see that the extreme outlying curves are all colored dark blue, which is a confirmation that our depth measure gives them a low depth value. We also notice some differences between Figure 4.9(a) and (b) around the time points 47ms and 57ms, which can be explained as in Section 4.3.1.



(a)



(b)



(c)

Figure 4.9: All curves colored according to their (a) univariate, and (b - c) bivariate MFHD depth.

We should however be cautious about this approach, as any data set, even one which only contains regular curves, will always indicate some of the curves as the ones with lowest depth. Moreover, as our functional depth measure averages the cross-sectional depth values it might give a large depth to a curve which is strongly outlying on part of its domain. Hence we recommend not

only to consider the global amount of outlyingness of a curve, measured by means of its MFHD, but also to consider its amount of local outlyingness. To this end, we reconsider the cross-sectional bivariate time points on which we have already computed the depth of each curve. As a by-product of these computations we can construct the bagplot, which is a bivariate extension of the boxplot [Rousseeuw et al., 1999a]. An example is given in Figure 4.10 at time $t = 57$ms. The bagplot draws a bag which contains the 50% curves with largest depth, and a fence which contains all the regular observations. Curves outside this fence can be flagged as outliers. We see that curves 112 and 207 are indeed flagged as being outlying at time $t = 57$, however both for a different reason. Curve 112 has an outlying acceleration value, whereas curve 207 has an outlying velocity. This can also clearly be seen from Figure 4.7.

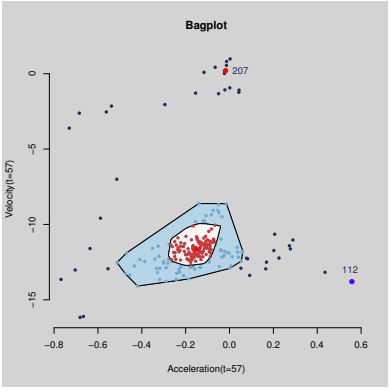
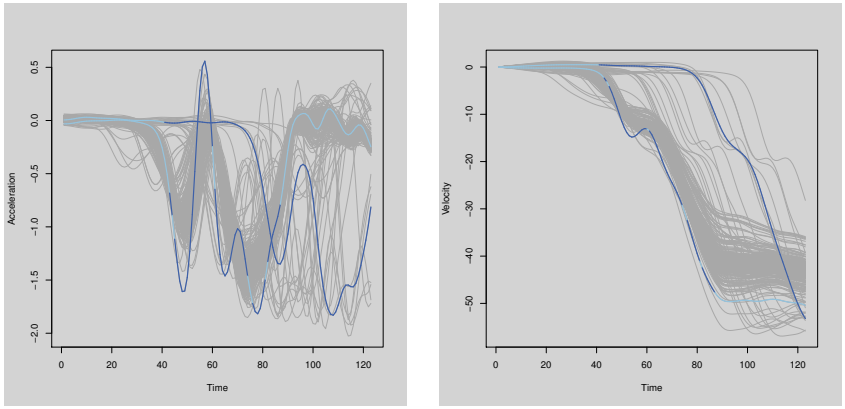


Figure 4.10: Bagplot at time point 57.

Next we can indicate for each curve at which time points it is flagged as a bivariate outlier. For curves 112 and 207, this is shown in Figure 4.11 where the dark blue parts of the curve indicate the regions where such a local outlyingness is detected (in contrast with the light blue parts where the curve belongs to the fence of the bagplot).

Finally we can compute for each curve the proportion of time points where it is marked as a local outlier. In Figure 4.12(a) we expose this proportion for all curves against their MFHD, which is more a global measure of outlyingness. We see that the curves with low MFHD also have many local regions of outlyingness. This provides more evidence that they really have an overall outlying behavior. On this plot we have added a vertical line through the 10% quantile of the MFHD values, and a horizontal line at 0.1. This clearly exposes the different types of curves. Those curves represented in the upper right corner are locally



(a) (b)

Figure 4.11: Local outlyingness for curves 112 and 207 shown on the (a) acceleration and the (b) velocity curves.

outlying in more than 10% of the time points but don't have an extremely low depth. This can be explained by the fact that MFHD also accounts for the amplitude variability whereas our local measure of outlyingness does not.

Note that this diagnostic display is currently limited to $K = 2$, as the bagplot is only defined for bivariate data. It is however very well suited for a nonparametric approach as the bagplot does not assume any parametric assumption about the data, apart from unimodality. A more powerful cross-sectional outlier identification procedure could of course be obtained if more assumptions (such as gaussianity) can be made.

Finally we compared our approach with the outliers found by the enhanced functional boxplot [Sun and Genton, 2012]. In that approach curves are globally flagged as outliers as soon as they exceed at some time point the fences, which are constructed based on the 50% central region as in the standard boxplot. First we derived the appropriate factor to inflate the central region as described in [Sun and Genton, 2012], which yielded the factor 1.5. The resulting outlying curves are indicated in Figure 4.12(b). We see that all flagged curves are also clearly visible in our diagnostic plot, either because their MFHD is very small, or because their proportion of local outlyingness is large. There are however some curves (36, 42 and 112) which are clearly outlying following our criteria, but which are not detected by means of the functional boxplot. The acceleration and velocity curves in Figure 4.12(c) and Figure 4.12(d) clearly show that these curves mainly have an outlying velocity behavior, which confirms our conclusion

based on MFHD. The functional boxplot on the other hand is only based on the acceleration curves and apparently was not able to detect these deviations.

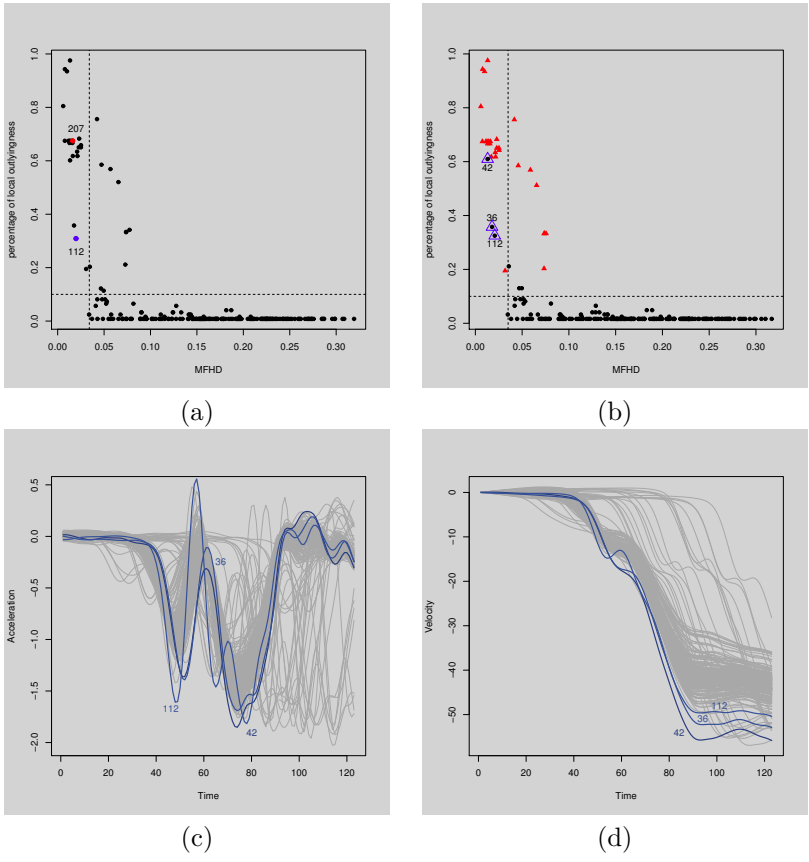


Figure 4.12: (a) Percentage of local outlyingness versus MFHD for all curves; (b) Same plot with outliers found by the enhanced functional boxplot indicated by red triangles; (c) acceleration and (d) velocity curves and some outlying curves according to MFHD.

4.3.2 U.K. weather data

In this section we present an example of bivariate functional data on which we illustrate the effect of using a different weight function. Our data set contains the

temperature and dewpoint temperature (in degrees Celsius) measured between January 11 and 17, 2013 at 78 weather stations in the U.K. The raw data were downloaded from NOAA (www.noaa.gov). The dewpoint temperature is the temperature at which air can no longer hold all of its enclosed water vapor. Some water vapor must then condensate into liquid water. It must be noted that the dewpoint temperature is always lower than the temperature. As different stations have different recording times, cubic spline interpolation was applied to obtain hourly estimates, yielding a total of 120 values, shown as the light grey curves in in Figure 4.13(a) and (b). Both temperatures clearly expose the day and night cycle. On these data we first applied MFHD with a constant weight function. Figures 4.13(c) and (d) show the resulting MFHD median and the boundaries of the 75% central regions by dashed curves.

Next we replaced eight curves with data from eight weather stations in Central Europe. They exhibit much lower temperatures as can be seen from the dark curves in Figure 4.13(a) and (b), and they affect the cross-sectional means heavily (depicted as solid lines). The solid lines in Figure 4.13(c) and (d) correspond to the MFHD median and the 75% central regions, computed on the contaminated data set. They are similar to the results based on the data from the U.K. only, which reflects the robustness of MFHD.

4.4 Simulations

In this section we present three simulation settings each designed to illustrate a particular aspect of the behavior of MFHD. In all cases, we generate $N = 50$ univariate curves $\{Y_1(t), \ldots, Y_N(t)\}$ from a stochastic process $\mathcal{Y}$, denoted as the uncontaminated curves $\{Y(t)\}_N$. Then we replace five curves of $\{Y(t)\}_N$ with curves sampled from a contaminating stochastic process $\mathcal{Y}_\varepsilon$, yielding a data set $\{Y_1^\varepsilon(t), \ldots, Y_N^\varepsilon(t)\} = \{Y_\varepsilon(t)\}_N$ with 10% contamination. All curves are evaluated on a grid of $T = 100$ equispaced time points $t_1, \ldots, t_T$ in $[0, 2\pi]$. Each experiment was replicated 100 times. In a first set of simulations (Section 4.4.2) we consider the bivariate MFHD applied to the curves and their derivatives. We compare its behavior with several univariate functional depths and with the bivariate random projection depth. Next, in Section 4.4.3 we illustrate the advantage of using the warping functions (or a function thereof) as additional curves. First we describe how we evaluate the performance and robustness of the functional depths.

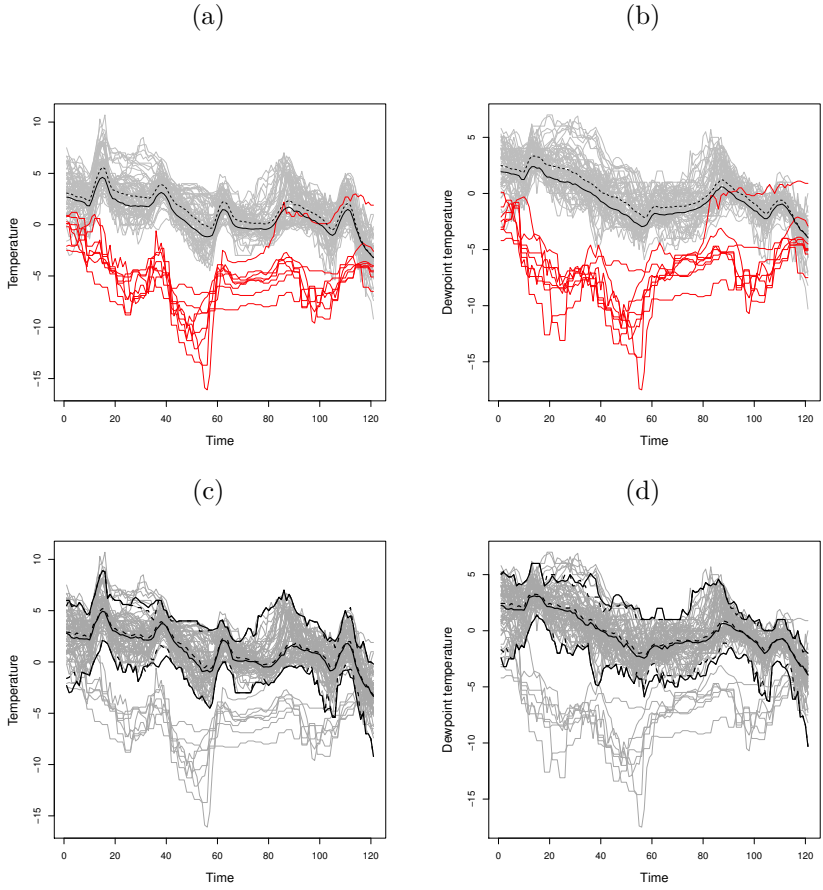


Figure 4.13: Weather data: (a)–(b) Temperature and dewpoint temperature for weather stations in the United Kingdom (light gray) and Central Europe (dark) with cross-sectional means of the U.K. data only (dashed curves) and cross-sectional means of the full dataset (solid curves) (c)–(d) MFHD median and central regions for the U.K. data only (dashed curves) and the full dataset (solid curves).

4.4.1 Evaluation criteria

ASE of the estimated central curve: the averaged squared scaled distance between the true and the estimated central curve,

$$\frac{1}{T}\sum_{j=1}^{T}\left(\frac{\widehat{m}_{Y_\varepsilon}(t_j)-m_{\mathcal{Y}}(t_j)}{s_{\mathcal{Y}}(t_j)}\right)^2,$$

where $m_{\mathcal{Y}}$ is the central curve of $\mathcal{Y}$, $\widehat{m}_{Y_\varepsilon}$ is the estimated central curve, and $s_{\mathcal{Y}}(t)$ is the interquartile range of $\mathcal{Y}(t)$.

ASE of the estimated dispersion curve: the average squared difference between the logarithm of the (0.5)-dispersion curves computed on the contaminated and the uncontaminated data,

$$\frac{1}{T}\sum_{j=1}^{T}\left(\log\left(\frac{s^{\varepsilon}_{0.5}(t_j)}{s_{0.5}(t_j)}\right)\right)^2,$$

where $s_{0.5}(t)$ is the width of the (0.5)-central region of $\{Y(t)\}_N$ as defined in Section 4.2.3. Analogously, $s^{\varepsilon}_{0.5}(t)$ is the dispersion curve computed from $\{Y_\varepsilon(t)\}_N$.

Normalized maximum depth of outliers. To have an easily comparable criterion, we normalize by dividing the maximum depth with the depth of the deepest curve. Formally, let $\mathrm{FD}_N(Y^{\varepsilon}_n, F_{\mathcal{Y}_\varepsilon,N})$ denote the (finite-sample) functional depth of the curves $Y^{\varepsilon}_n(t)$ from $\{Y_\varepsilon(t)\}_N$, and denote by I_c the index set of the contaminated curves from $\{Y_\varepsilon(t)\}_N$. Then we consider $\max_{n\in I_c}\mathrm{FD}_N(Y^{\varepsilon}_n, F_{\mathcal{Y}_\varepsilon,N})/\max_{n=1,\dots,N}\mathrm{FD}(Y^{\varepsilon}_n, F_{\mathcal{Y}_\varepsilon,N})$.

4.4.2 Simulations with curves and their derivatives

We generate three types of univariate curves, contaminated with 10% outlying curves, and we evaluate MFHD on the bivariate samples $\{(Y_\varepsilon(t), Y'_\varepsilon(t))\}_N$. We consider both $\alpha = 1/4$ and $\alpha = 1/8$, resulting in MFHD(1/4) and MFHD(1/8). In both cases the estimated central curve $\hat{m}_{Y_\varepsilon}(t)$ is the MFHD median.

We compare the behavior of MFHD on the bivariate samples, first, with three approaches applied on the univariate curves $\{Y_\varepsilon(t)\}_N$. (1) The cross-sectional average (CSA): $\widehat{m}_{Y_\varepsilon}(t) = \frac{1}{50}\sum_{n=1}^{50} Y^{\varepsilon}_n(t)$. The corresponding depth is the univariate Mahalanobis depth, with $\widehat{\sigma}_{Y_\varepsilon}(t)$ the cross-sectional standard

deviation,

$$\mathrm{MD}(Y_n^\varepsilon(t)) = \left\{1 + \left(\frac{Y_n^\varepsilon(t) - \widehat{m}_{Y_\varepsilon}(t)}{\widehat{\sigma}_{Y_\varepsilon}(t)}\right)^2\right\}^{-1}.$$

(2) The modified band depth (MBD) of [López-Pintado and Romo, 2009]. This corresponds with $\mathrm{MFD}(Y_i, \mathcal{Y}_\varepsilon)$ as in (4.3) with the simplicial depth [Liu, 1990] as depth function D and with constant weight function $w(t) = 1/T$. The curve with largest MBD is considered as $\widehat{m}_{Y_\varepsilon}(t)$. We use the implementation provided in the R package `depthtools`. (3) The MFHD($\alpha = 1/4$) applied to the univariate curves $\{Y_\varepsilon(t)\}_N$, which we denote by UFHD(1/4). Note that the deepest curve in this case corresponds with the cross-sectional median.

Next, we compare with the bivariate random projection depth (RPD) of [Cuevas et al., 2007], using the default settings from the implementation in the R package `fda.usc` [Febrero-Bande and Oviedo de la Fuente, 2012]. Here, the curves and their derivatives are projected on a random direction, yielding a bivariate sample on which the (bivariate) modal depth [Cuevas et al., 2006] can be computed for all observations. The RPD of a curve corresponds with the average modal depth over 50 random projections. Note that this approach does not satisfy the affine invariance property as stated in Theorem 1.

The functional derivatives are computed using B-splines using the default settings and the algorithms from the R-package `fda.usc`.

Simulation setting I: Shifted outliers

This simulation setting illustrates the behavior of MFHD in cases where the true curves are homoscedastic and all derivative curves follow the same process. We generated curves of the form

$$\begin{aligned} Y_n^\varepsilon(t) &= (1 - c_n)\{a_{1n}\sin(t) + a_{2n}\cos(t)\} \\ &\quad + c_n\{a_{1n}\sin(t) + a_{2n}\cos(t) + \frac{1}{4}\}, \end{aligned}$$

where t is a grid of 100 equispaced values on $[0, 2\pi]$, c_n is 1 for 10% of the curves and 0 otherwise. The random coefficients a_{1n} and a_{2n} follow independent uniform distributions on $[0, 0.05]$. Figure 4.14(a) depicts the 'regular' (solid) and outlying (dashed) curves and Figure 4.14(d) the corresponding derivatives.

The first panel in Figure 4.15 depicts, for each of the functional depth methods, the ASE of the central curves. CSA is highly influenced by the outlying curves,

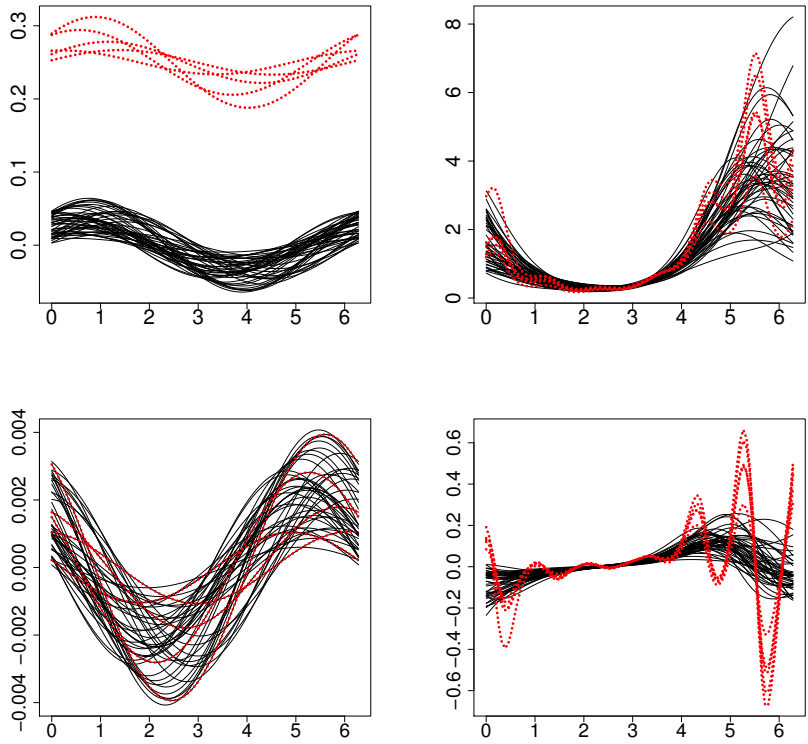


Figure 4.14: Simulation settings. Main curves (solid) and outliers (dashed) for the curves (a),(b) and their derivatives (c),(d) for (a),(c) the shifted outliers; (b),(d) the log-normal processes.

and RPD to a lesser extent. The middle panel in Figure 4.15 depicts the ASE of the dispersion curve. The effects of the outliers on the CSA estimate of dispersion $s_{0.5}(Y_n^{\varepsilon}(t))$ is more muted because the outlying curves are located too far to be included in the set of 25 curves with largest Mahalanobis depth. RPD contains some large values too. All other methods perform well on both criteria. The third panel in Figure 4.15 depicts the (normalized) maximum depth of outliers. Here again, CSA assigns a high depth to the outliers. Both univariate functional depths (MBD and UFHD) assign higher depths to the outliers than the bivariate depth functions. The value of α for MFHD(α) has a negligible effect on all three performance criteria.

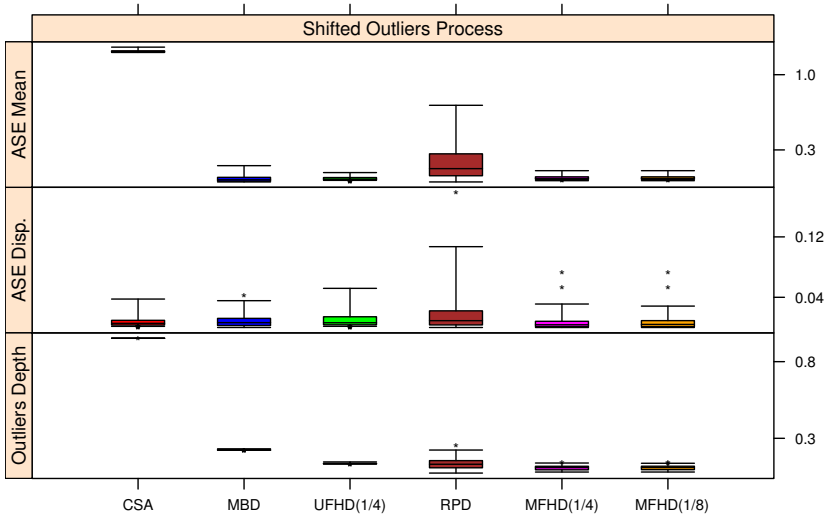


Figure 4.15: Setting I: shifted outliers. ASE of the estimated central curve (top), of the 0.5-dispersion curve (middle) and outlier depth (bottom).

Simulation setting II: Log-normal processes

Highly heteroscedastic curves are obtained by generating from a log-normal process $Y_n(t) \sim \log N(\mu(t), \Sigma(t))$. Denote $\boldsymbol{x} = \{x_i\}_{i=1}^{20}$ 20 equidistant points on $[0, 2\pi]$. The covariance kernel of the x_i's is given by $K_{xx}(i,j) = \exp\left(-\frac{(x_j - x_i)^2}{2\delta^2}\right)$, with $\delta = 0.25$. For the time points $\boldsymbol{t} = \{t_j\}_{j=1}^{100}$ equidistant on $[0, 2\pi]$, we define K_{tx} and K_{tt} analogously. Then, the weight matrix for the mean μ is $K^m = K_{tx}(K_{xx} + D)^{-1}$ where D is a diagonal matrix that directs the heteroscedasticity of the final process, $D = \text{Diag}_{i=1,\ldots,20}\{\min((\pi - x_i)^2, 1)\}$, so that the variance of the process is minimized at $t = \pi$. For the regular curves we take $\boldsymbol{\mu}(\boldsymbol{t}) = K^m(a_1 \sin(\boldsymbol{x}) + a_2 \cos(\boldsymbol{x}))$, where $a_1 \sim \mathcal{U}(-2, 2)$ and $a_2 \sim \mathcal{U}(-1, 1)$ are randomly generated. For the outlying curves we took $\boldsymbol{\mu}^*(\boldsymbol{t}) = K^m(\sin(6\boldsymbol{x}) + \boldsymbol{\mu}(\boldsymbol{x}))$. The covariance matrix is given by $\Sigma = K_{tt} - K_{tx}(K_{xx} + D)^{-1}K'_{tx}$. For a generated sample of curves and the corresponding derivatives, see Figure 4.14(c),(f).

This configuration was designed to penalize those estimators that do not use the information from the derivatives of the curves to assign depths. This is particularly visible in the third panel of Figure 4.16, where the CSA, MBD,

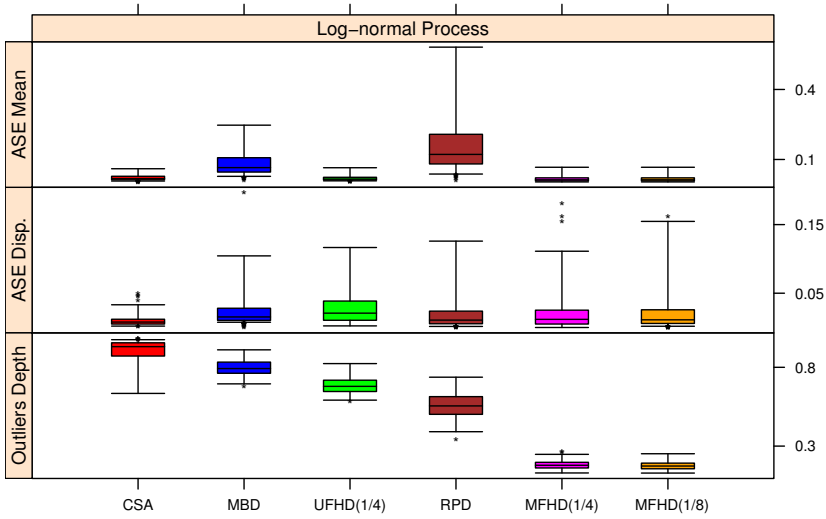


Figure 4.16: Setting II: log-normal processes. ASE of the estimated central curve (top), of the 0.5-dispersion curve (middle) and outlier depth (bottom).

UFHD and RPD are unable to detect the outlying curves. The outliers do not affect the CSA in terms of ASE of the central and the dispersion curves since the range of the response values is the same for all curves. Although RPD uses the derivatives, it does not perform well; RPD is not estimating the central curve well, and it does not assign low depths to the outlying curves. MFHD retains its good behavior.

4.4.3 Simulation with warped curves

Warping can make outlying curves more difficult to detect by pulling them towards the uncontaminated ones. See Figure 4.17 for an example where outlying curves are initially visible but are then hidden by the warping process. Here, using a bivariate approach can help with the ranking of the curves. For MFHD we compare two bivariate approaches. First, we create a bivariate sample of curves by using the warped curves together with the individual warping functions. Second, we use as a bivariate sample of curves the warped curves together with the derivatives of the warping functions. Adding the curves related to warping alleviates the information loss induced by the warping procedure.

For the warping functions, our simulation design follows that used in the first simulation setting of [Arrabis-Gil and Romo, 2012]. The warping functions for the good curves are generated as explained on their page 405, formula (3.7). The inverse of

$$h_n(t) = \frac{\arctan(\beta_n(2t-1))}{2\arctan(\beta_n)} + 1/2,\ t \in [0,1]$$

with β_n equally spaced between 10 and 14 is used as warping function for the outliers. We use the same amplitude functions for all curves such that different warping functions correspond to original curves with different shapes and phases. To make the results comparable with those of simulation setting I, we use $Y_n(t) = a_{1i}\sin(t/(2\pi)) + a_{2i}\cos(t/(2\pi))$. A sample of curves and warped curves is depicted in Figure 4.17, together with the warping functions and their derivatives.

For comparison we use two versions for UFHD: only the unwarped curves, and only the warped curves.

Figure 4.18 contains a summary of the performance criteria. As expected, once warped, the outlying curves have no influence on the estimation of the central (or dispersion curves) and this is visible in the first two panels of Figure 4.18 where UFHD has low MSE on both measures. At the same time, warping makes the curves with different shapes similar to the other curves, causing a poor behavior of UFHD on the warped curves in the third panel. Adding the warping functions, or their derivatives in a bivariate analysis completely addresses the information loss introduced by the warping procedure.

4.5 Discussion

We have presented a new depth function for multivariate functional data (MFD), defined as a weighted average of the cross-sectional multivariate depths. It assigns a ranking to curves from the center outwards, whilst accounting for differences in amplitude. Shape and phase variation can be accommodated by including derivatives and/or warping functions. Interesting theoretical properties and computational advantages are achieved when using the multivariate halfspace depth, which leads to the MFHD depth function. Notably, the population counterpart of MFHD satisfies the four properties listed in [Zuo and Serfling, 2000], i.e. affine invariant, maximal at the center, monotone relative to the deepest point and vanishing at infinity. Furthermore, at unimodal elliptic symmetric distributions, the contours of the depth regions coincide with density contours, which implies that the choice of α in MFHD(α) becomes irrelevant at the population level. Additional advantages of the halfspace depth

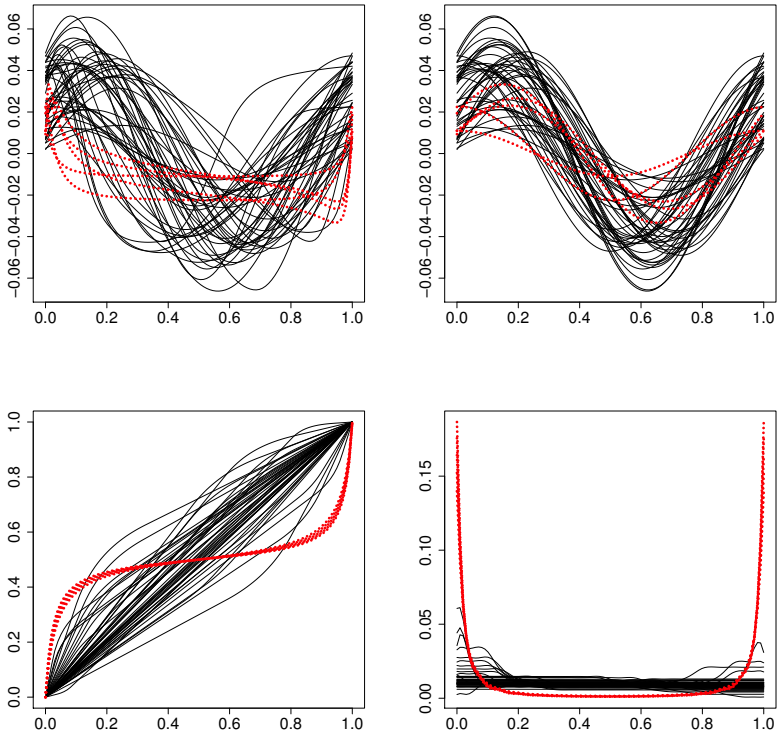


Figure 4.17: Simulation setting III. (Top Left) Original curves, (Top Right) warped curves, (Bottom Left) warping functions, (Bottom Right) derivatives of the warping functions; outlying curves as dashed lines.

are its robustness with respect to outliers and the existence of fast algorithms for computing it as well as the corresponding depth regions and Tukey median. The multivariate functional median curve can then be computed explicitly and estimates the central behavior of the curves. MFHD also allows to visualize and quantify the variability amongst the curves. Simulations have shown the benefit of adding derivatives or warping information to univariate curves, and they have illustrated the better performance of MFHD compared with the bivariate random projection depth.

As illustrated in the data examples and in the simulations, MFHD assigns lower depth to curves which deviate strongly from the majority of the curves.

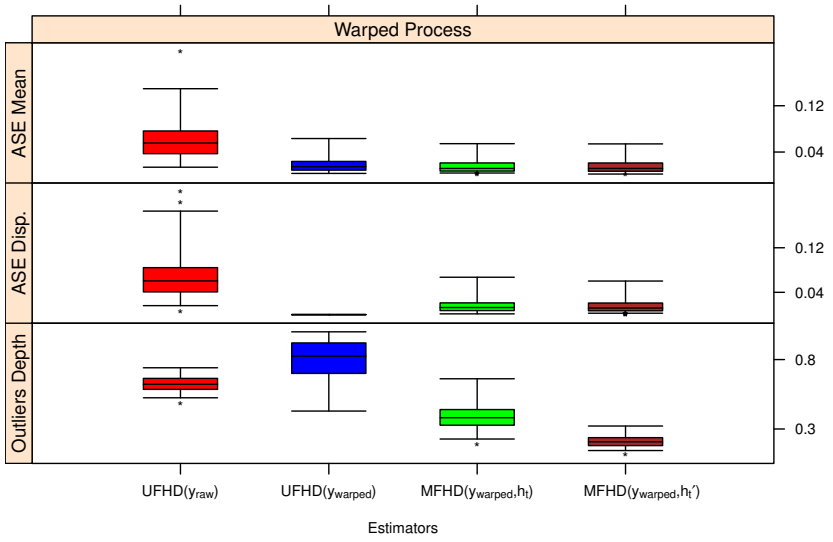


Figure 4.18: Setting IV. ASE of the estimated central curve (top) and the 0.5-dispersion curve (middle) and outlier depth (bottom). Using the original curves (y_{raw}), the warped curves (y_{warped}) and their warping functions (h_t) with derivatives h_t'.

This robustness towards outliers is inherited from the halfspace depth which is applied at every time point. It is however well known that other depth functions, such as projection depth, attain a higher breakdown value and thus could lead to a more robust MFD. Alternatively one could replace the average in (4.3) by an infimum as proposed in [Mosler, 2013]. More theoretical and numerical studies are needed to compare these different depth functions.

Chapter 5

The mrfDepth package

5.1 Introduction

The concept of depth first appeared in the context of multivariate data when [Tukey, 1975] introduced halfspace depth. Since then several different notions of depth have been introduced including the simplicial depth by [Liu, 1988, Liu, 1990], the projection depth by [Zuo and Serfling, 2000] and adjusted outlyingness by [Hubert and Van der Veeken, 2008]. Depth is now recognised as a way to order the data from the center outwards. The notion of depth however is not limited to multivariate data. Regression depth, for example, was introduced by [Rousseeuw and Hubert, 1999]. Since then new data types have emerged leading to new fields of interest in statistics.

One such emerging data type is functional data. Functional data comprises of measurements depending on continuous parameters. One can think of spectra in chemometrics that depend on a frequency variable or measurements taken over time. Standard reference work on functional data analysis includes [Ramsay and Silverman, 2002, Ramsay and Silverman, 2005] and [Ferraty and Vieu, 2006]. The methodology of functional data has been applied in several fields such as image processing by [Ogden et al., 2002] and in medicine by [Pfeiffer et al., 2002] and by [Ieva et al., 2012].

The notion of depth corresponding to this type of data followed suit and several depth functions have been proposed for this new type of data. One of the first was the Fraiman and Muniz depth proposed by [Fraiman and Muniz, 2001] for samples of continuous functions. Further developments were made by [Cuevas et al., 2007] proposing the random projection depth for data belonging

to the Hilbert Space $L[0,1]$. [López-Pintado and Romo, 2009] introduced band and modified band depth and half-region and modified half-region depth [Lopez-Pintado and Romo, 2011]. Recently [Claeskens et al., 2014] proposed multivariate functional depth. An important factor in the success of these ideas is the availability of implementations in open source, widely used, software suites.

Several of these depth functions are implemented in `R` [R Core Team, 2014]. The `depth` package by [Genest et al., 2012] bundles implementations for the calculation of halfspace depth, simplicial depth and Oja depth in the multivariate setting. Implementations for the projection depth and the adjusted outlyingness can be found in the R packages `robust` and `rrcov` by [Wang et al., 2013] and [Todorov and Filzmoser, 2009]. The Fraiman and Muniz depth and the random projection depth can be calculated in the R package `fda.usc` by [Febrero-Bande and Oviedo de la Fuente, 2012]. The modified band depth has been implemented in the R package `depthTools` by [Lopez-Pintado and Torrente, 2013] and the multivariate functional halfspace depth has been implemented [Claeskens et al., 2014] in the package `MFHD`. For MATLAB [MATLAB, 2014] the `LIBRA` toolbox by [Verboven and Hubert, 2005] implements bivariate halfspace depth. Depth based notions such as the bagplot and bivariate halfspace contours are available in the R packages `aplpack` by [Wolf and Bielefeld, 2013] and the `depth` package as well as the `LIBRA` toolbox for `MATLAB`.

With the `mrfDepth` package we want to present several depth notions in a unified framework as well as making available implementations of the regression depth and multivariate functional depth whilst offering several improvements over existing implementations in the referenced packages. These improvements were made in therms of usability, performance and available options. Detailed descriptions of them are given in the corresponding sections throughout the chapter. Furthermore the proposed software is made available for both R and MATLAB in a way that is cross-platform consistent. Finally the package implements several graphical tools that allow for graphical interpretation and inspection of multivariate and functional data based on depth.

In the following sections the three different data types are treated separately. In Section 5.2 multivariate depth is discussed. Sections 5.3 and 5.4 discuss regression and functional depth respectively.

5.2 Multivariate Depth and Outlyingness

The `mrfDepth` package includes both updated and new implementations of some well known estimators for depth and outlyingness-based exploration and

inference for multivariate data. In this section we briefly recapitulate these estimators and discuss their implementations. We begin by the various ones built around the concept of outlyingness of an observation with respect to a cloud of points.

Given a data set $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ where the rows (denoted $\{\boldsymbol{x}_i\}_{i=1}^n$) represent the observations and the columns are the measurements and a p-dimensional vector $\boldsymbol{z}$, an outlyingness function $O(\boldsymbol{z}, \boldsymbol{X})$ is a center-outward ordering of the points $\boldsymbol{z} \in \mathbb{R}^p$ with higher values associated with those points $\boldsymbol{z}$ that are in some sense "far out" from the center of $\boldsymbol{X}$.

Perhaps the simplest concept of multivariate outlyingness is that of projection outlyingness [Stahel, 1981, Donoho, 1982]. Projection outlyingness is based on the geometrical insight that any multivariate outlier should also stand out on at least one univariate projection of the data. The Stahel-Donoho outlyingness of $\boldsymbol{z}$ with respect to $\boldsymbol{X}$ is defined as:

$$\mathrm{PO}(\boldsymbol{z}, \boldsymbol{X}) = \sup_{\boldsymbol{a} \in \mathbb{R}^p : ||\boldsymbol{a}|| = 1} \frac{|\boldsymbol{a}^\top \boldsymbol{z} - \hat{\mu}(\boldsymbol{X}\boldsymbol{a})|}{\hat{\sigma}(\boldsymbol{X}\boldsymbol{a})}. \tag{5.1}$$

Typically $\hat{\mu}$ is chosen to be the median and $\hat{\sigma}$ to be the MAD, but our implementation also offers a choice for setting $(\hat{\mu}, \hat{\sigma}) = (\hat{\mu}_{\mathrm{MCD}}, \hat{\sigma}_{\mathrm{MCD}})$ where the latter are the univariate MCD location and scale estimators [Rousseeuw and Leroy, 1987]. These estimators are defined as the mean and standard deviation of the $[n/2] + 1 \leqslant h \leqslant n$ observations with smallest variance and can be computed in $O(n \log n)$ time. Because it is intractable to consider all directions in $\mathbb{R}^p$, one often substitutes Equation (5.1) by the computable alternative:

$$\mathrm{PO}(\boldsymbol{z}, \boldsymbol{X}) = \sup_{\boldsymbol{a} \in \mathcal{B}} \frac{|\boldsymbol{a}^\top \boldsymbol{z} - \hat{\mu}(\boldsymbol{X}\boldsymbol{a})|}{\hat{\sigma}(\boldsymbol{X}\boldsymbol{a})} \tag{5.2}$$

where $\mathcal{B}$ is a suitable set of directions. Often, $\mathcal{B}$ will be the set of all directions perpendicular to hyperplanes through p data points from $\boldsymbol{X}$. This particular choice of $\mathcal{B}$ (which we will denote $\mathcal{B}_p(\boldsymbol{X})$) is computational expensive but has the advantage of rendering the resulting vectors of outlyingness affine invariant. This means that:

$$\mathrm{PO}(\boldsymbol{z}, \boldsymbol{X}) = \mathrm{PO}(\boldsymbol{A}\boldsymbol{z}, \boldsymbol{A}\boldsymbol{X}) \tag{5.3}$$

for any non-singular $p \times p$ matrix $\boldsymbol{A}$. Note that the affine invariance of $\mathrm{PO}(\boldsymbol{z}, \boldsymbol{X})$ is preserved if we restrict ourselves to a random subset of elements from $\mathcal{B}_p(\boldsymbol{X})$.

In the `mrfDepth` package, the `C++` code computing the projection outlyingness is callable through the `outlyingness` function. The Stahel Donoho estimator [Maronna and Yohai, 1995] is already widely implemented, notably in the `R` package `rrcov` [Todorov and Filzmoser, 2009]. Nonetheless, the new code we propose improves on existing ones in several respects. Firstly, it is written in a native and object oriented programing language (`C++`) and uses standardised and open source library functions as its basic building blocks (most notably the Eigen `C++` library for linear algebra, [Guennebaud et al., 2013]). This ensures that the resulting code is not only shorter and easier to use within other applications but also runs much faster. For example, compared to the `Fortran` implementation of SDE in the `rrcov` package, we find an average speed up of up to 500% (when both n and p are large).

Secondly, our new implementation lets the user choose alternative sets of directions to sample from beside $\mathcal{B}_p(\boldsymbol{X})$. This option is motivated by the fact that in some settings, most notably situations when $n < p$, directions perpendicular to hyperplanes through p data points (and hence the elements of $\mathcal{B}_p(\boldsymbol{X})$) are not uniquely defined. In those cases, it can be interesting to consider the set $\mathcal{B}_2(\boldsymbol{X})$ of directions through pairs of data points from $\boldsymbol{X}$ instead (for example, this approach is used in the first step of the ROBPCA algorithm [Hubert et al., 2005]). Computing the outlyingness on projection of the data onto members of $\mathcal{B}_2(\boldsymbol{X})$ rather than members of $\mathcal{B}_p(\boldsymbol{X})$ is also much more tractable computationally because the cost of obtaining a single projection scales as $O(p)$ –rather than $O(p^3)$. However, substituting the set $\mathcal{B}_p(\boldsymbol{X})$ by $\mathcal{B}_2(\boldsymbol{X})$ also causes the resulting outlyingness index to no longer be affine invariant but only orthogonal invariant. This means that:

$$\mathrm{PO}(\boldsymbol{z}, \boldsymbol{X}) = \mathrm{PO}(\boldsymbol{A}\boldsymbol{z}, \boldsymbol{A}\boldsymbol{X}) \tag{5.4}$$

for any $p \times p$ matrix $\boldsymbol{A}$ for which $\boldsymbol{A}^\top = \boldsymbol{A}^{-1}$. The orthogonal equivariance of the squared outlyingness is preserved if we restrict ourselves to a random subset of fixed size of elements from $\mathcal{B}_2(\boldsymbol{X})$. Thirdly, while the `rrcov` implementation of the SDE only allows the user to consider a random sample of directions from $\mathcal{B}_p(\boldsymbol{X})$ we offer the possibility to also carry (when this is computationally possible, that is when n and p are small enough) an exhaustive search over all the members of $\mathcal{B}_p(\boldsymbol{X})$ (the same option is also offered for $\mathcal{B}_2(\boldsymbol{X})$, though in this case the computational cost of doing so only depends on n). This has the effect that the resulting outlyingness index is no longer stochastic but becomes fully deterministic. Finally, we allow the user to feed the algorithm with points for which the SD-outlyingness has to be computed but which are not used to compute the estimates themselves. The availability of this option is particularly important in cases when the SDE is used so asses the outlyingness of incoming observations. In the `mrfDepth` package, this computation is performed by

the `outlyingness` function. For example, the code below loads the `mrfDepth` package, creates a new random data matrix object named `X` and computes the projection outlyingness of the rows (observations) of `X`.

```
R> library("mrfDepth")
R> X <- matrix(rnorm(100 * 3) , nc = 3)
R> outlyingness_output <- outlyingness( X )
```

Another function implemented in the `mrfDepth` package is `adjOutlyingness`. It computes the Adjusted Outlyingness of a dataset. The Adjusted outlyingness itself was introduced in [Brys et al., 2005] and studied in detail by [Hubert and Van der Veeken, 2008] and can be seen as a generalization to multivariate skewed distributions of the Stahel-Donoho outlyingness. The Adjusted Outlyingness is built around the Medcouple [Brys et al., 2004], a robust measure of skewness. For a n-vector $\boldsymbol{x}$ with no ties, the Medcouple is defined as:

$$\mathrm{MC}(\boldsymbol{x}) = \underset{x_i < \operatorname{med}\boldsymbol{x} < x_j}{\operatorname{med}} \frac{(x_j - \operatorname{med}\boldsymbol{x}) - (\operatorname{med}\boldsymbol{x} - x_i)}{x_j - x_i} \tag{5.5}$$

Then, for directions $\boldsymbol{a} : \mathrm{IQR}(\boldsymbol{X}\boldsymbol{a}) > 0$, where for a given n-vector X $Q_1(X)$ and $Q_3(X)$ stand for the first and third quartile of the entries of X and $\mathrm{IQR}(X) = Q_3(X) - Q_1(X)$. The Adjusted Outlyingness is defined as:

$$\mathrm{AO}(\boldsymbol{z}, \boldsymbol{X}) = \sup_{\boldsymbol{a} \in \mathbb{R}^p} \begin{cases} \frac{\boldsymbol{a}^\top \boldsymbol{z} - \operatorname{med} \boldsymbol{X}\boldsymbol{a}}{c_2(\boldsymbol{X}\boldsymbol{a}) - \operatorname{med} \boldsymbol{X}\boldsymbol{a}} & \text{if } \boldsymbol{a}^\top \boldsymbol{z} > \operatorname{med} \boldsymbol{X}\boldsymbol{a} \\ \frac{\operatorname{med} \boldsymbol{X}\boldsymbol{a} - \boldsymbol{a}^\top \boldsymbol{z}}{\operatorname{med} \boldsymbol{X}\boldsymbol{a} - c_1(\boldsymbol{X}\boldsymbol{a})} & \text{if } \boldsymbol{a}^\top \boldsymbol{z} < \operatorname{med} \boldsymbol{X}\boldsymbol{a} \end{cases} \tag{5.6}$$

where for an n-vector X, $c_1(X)$ is the smallest observation greater than $Q_1(X) - 1.5 \exp(-3.5\mathrm{MC}(X))\mathrm{IQR}(X)$ and $c_2(X)$ is the largest observation smaller than $Q_3(X) + 1.5 \exp(4\mathrm{MC}(X))\mathrm{IQR}(X)$. The actual computation of the Medcouple is done using the fast and deterministic algorithm also introduced in [Brys et al., 2004]. In practice, this means that even for large data sets, the AO can be computed for a multiple (of order $O(\log n)$) of the cost of computing the SDE. As with the SDE, it is often impractical to compute the AO along all directions $\boldsymbol{a} \in \mathbb{R}^p$ but, depending on the context, it will often be sufficient to only consider directions $\boldsymbol{a} \in \mathcal{B}_p(\boldsymbol{X})$ or even $\boldsymbol{a} \in \mathcal{B}_2(\boldsymbol{X})$.

In the `mrfDepth` package, the `C++` code computing the adjusted outlyingness is callable through the `adjOutlyingness` function. The Adjusted outlyingness has previously been implemented in the `LIBRA` toolbox (for MATLAB®) and the `robustbase` package (for `R`). Nonetheless, the new code improves on existing

ones in several respects. Firstly, we have implemented the algorithm in a native language, `C++` , resulting in considerable speed up over existing `R` and MATLAB® implementations. To fix ideas, in tests, we find that these speed ups vary between two and three orders of magnitude depending on the values of n (the number of observations) and p (the number of measurements). Secondly and as in our implementation of SDE, we now allow the user to determines the set of direction used in the computations of AO. Paralleling the options available for the SDE, our implementation also gives the option to the end user to compute the AO exhaustively by considering all members of $\mathcal{B}_p(\boldsymbol{X})$ (or $\mathcal{B}_2(\boldsymbol{X})$), making the resulting AO index deterministic. Finally, as with our implementation of SDE, we allow to user to specify the $\boldsymbol{z}$ (those points for which the AO are to be computed) separately from the $\boldsymbol{X}$ (those points used in the computation of the AO).

An alternative measure of localization of a point $\boldsymbol{z}$ with respect to a (potentially multivariate) sample $\boldsymbol{X}$ is that of depth. As with the concept of outlyingness, a depth functional $D(\boldsymbol{z}, \boldsymbol{X})$ provides a center-outward ordering of the point in $\mathbb{R}^p$. The main difference lies with the convention that larger values of $D(\boldsymbol{z}, \boldsymbol{X})$ are associated with points $\boldsymbol{z}$ that, in some sense, lie "deeper" inside the point cloud formed by the members of $\boldsymbol{X}$ [Zuo and Serfling, 2000]. Indeed, the concepts of depth and outlyingness are strongly related to one another. For example, given the projection (resp. adjusted) outlyingness of an point $\boldsymbol{z}$ with respect to a dataset $\boldsymbol{X}$, one can always compute the associated projection depth (resp. adjusted projection depth) value through the relations:

$$\mathrm{PD}(\boldsymbol{z}, \boldsymbol{X}) = \frac{1}{1 + \mathrm{PO}(\boldsymbol{z}, \boldsymbol{X})} \tag{5.7}$$

$$\mathrm{APD}(\boldsymbol{z}, \boldsymbol{X}) = \frac{1}{1 + \mathrm{AO}(\boldsymbol{z}, \boldsymbol{X})} \tag{5.8}$$

In the `mrfDepth` package, this computation is performed by the `projDepth` function. For example, continuing with the example introduced above:

```
R> adjOutlyingness_output <- adjOutlyingness( X )
R> projDepth(adjOutlyingness_output)
```

The code for calling `projDepth` on the result of a call to `outlyingness` is nearly similar:

```
R> Outlyingness_output <- outlyingness( X )
```

```
R> projDepth(Outlyingness_output)
```

The `mrfDepth` package also contains implementations of several depth based multivariate estimation and data exploration tools. Below, we briefly describe these. We begin with the `hdepth` function, a wrapper to the `Fortran` implementation of the fast and deterministic algorithms of [Rousseeuw and Ruts, 1996] (for the case $p = 2$) and [Rousseeuw and Struyf, 1998] (for the case $p = 3$) and the fast and approximate algorithm of [Rousseeuw and Struyf, 1998] (for the case when $p > 3$). These algorithms compute the halfspace depth of $\boldsymbol{z}$ with respect to $\boldsymbol{X}$ which is defined as [Tukey, 1975]:

$$\mathrm{HD}(\boldsymbol{z}, \boldsymbol{X}) = \inf_{\boldsymbol{a} \in \mathbb{R}^p : ||\boldsymbol{a}|| = 1} \frac{1}{n} \#\{i : \boldsymbol{x}_i^\top \boldsymbol{a} \geqslant \boldsymbol{a}^\top \boldsymbol{z}\}. \tag{5.9}$$

The `Fortran` implementation of these algorithm were already accessible through the `R` package `depth` [Genest et al., 2012] but we enhanced their usability in several ways. Firstly, our implementation also accepts matrices $\boldsymbol{Z} = \{z_i\}_{i=1}^m$ as inputs (as opposed to individual p-vectors). We have added a check for exact fit situations (whereby all the vectors of $\boldsymbol{X}$ collapse to a subspace) and we now perform the computations on standardized inputs. More precisely, these routines now run on $\boldsymbol{Z}^*$ and $\boldsymbol{X}^*$ (instead of the original $\boldsymbol{Z}$ and $\boldsymbol{X}$) where:

$$\begin{aligned} x_{ij}^* &= \frac{x_{ij} - \mathrm{ave}_i(x_{ij})}{\mathrm{sd}_i(x_{ij})} \\ z_{ij}^* &= \frac{z_{ij} - \mathrm{ave}_i(x_{ij})}{\mathrm{sd}_i(x_{ij})} \end{aligned}$$

We also added these improvements (re-scaling the inputs, detection of exact fits) to the `bagplot`, `isohdepth`, `sdepth` and `maxhdepth` functions which we discuss now. Since the vector of depth returned by all these algorithms are affine invariant, performing this initial standardization does not affect their theoretical properties. Nonetheless, in numerous tests, we have found that this step improves numerical stability and hence we include it by default.

When $p = 2$, the function `sdepth` computes the simplicial depth [Liu, 1988] of $\boldsymbol{z}$ with respect to $\boldsymbol{X}$. The simplicial depth of a point with respect to a bivariate sample is the number of triangles formed by points of $\boldsymbol{X}$ that contain $\boldsymbol{z}$, divided by a factor $\binom{n}{3}$ to account for the total number of such triangles in an n by p data matrix $\boldsymbol{X}$. The `sdepth` function is a wrapper to the fast algorithm of [Rousseeuw and Ruts, 1996] which computes the simplicial depth in time $O(n \log n)$.

The `isohdepth` function is a wrapper for the algorithm of the same name [Ruts and Rousseeuw, 1996]. Given a bivariate data set $\boldsymbol{X}$, the isodepth algorithm computes the contours (and the volumes) of the depth region D_α, $(0 < \alpha < 1)$ efficiently. The depth region D_α is the set of points for which $\mathrm{HD}(\boldsymbol{z}, \boldsymbol{X}) \geqslant \alpha$. The depth regions are bounded, convex sets nested for increasing α. Contrary to the wrapper to `isohdepth` found in the `R` package `depth`, ours only calls the more recent [Rousseeuw et al., 1999a] implementation of the algorithm.

The `mrfDepth` package also contains `maxhdepth`, a wrapper for various algorithms [Rousseeuw and Ruts, 1998] designed to compute the Tukey median of a cloud of points. Given a cloud of points $\boldsymbol{X}$, the Tukey median of $\boldsymbol{X}$ is the point $\boldsymbol{z}^+$ having largest halfspace depth with respect to $\boldsymbol{X}$. Often, the point $\boldsymbol{z}^+$ will not be unique in which case the Tukey median is the average of the set of points $\boldsymbol{Z}^+ = \{\boldsymbol{z}_i^+\}$ with maximal halfspace depth. For any dataset $\boldsymbol{X}$ in *general position* (meaning that the space generated by any $p+1$ vector in $\boldsymbol{X}$ equals $\mathbb{R}^p$), it always holds that [Donoho and Gasko, 1992]:

$$\frac{\left\lceil \frac{n}{p+1} \right\rceil}{n} \leqslant \max_{\boldsymbol{z}} \mathrm{HD}(\boldsymbol{z}, \boldsymbol{X}) \leqslant \frac{\left\lceil \frac{n}{2} \right\rceil}{n} \tag{5.10}$$

Here, when $p \leqslant 3$ the main change with respect to the wrapper found in the `R` package `depth` is that the algorithm now calls the `hdepth` and `isohdepth` routines in the `Fortran` code whenever possible (avoiding many duplications in the `Fortran` code). When $p > 4$ there are no fast, exact algorithm for computing the halfspace depth of a cloud of points. Here, our package implements the [Cuesta-Albertos and Nieto-Reyes, 2008] algorithm, yielding significant speed ups over the [Rousseeuw and Struyf, 1998] algorithm implemented in the `R` package `depth`: in tests, we found between 1 (when both n and p are small) and 3 (when both n and p are large) orders of magnitude reduction in running times.

Finally, the `mrfDepth` package also contains an implementation of the [Rousseeuw and Struyf, 2002] multivariate depth based test of symmetry. This is a rank test for the null hypothesis that a multivariate sample is drawn from a continuous distribution F which is angularly symmetric about a location vector $\boldsymbol{\theta}_0$. Angular symmetry is a broadening of the concept of central symmetry first introduced by [Liu, 1988]. A random vector $\boldsymbol{x}$ has an angularly symmetric distribution about $\boldsymbol{\theta}$ if

$$\frac{\boldsymbol{x} - \boldsymbol{\theta}}{||\boldsymbol{x} - \boldsymbol{\theta}||} \overset{d}{=} \frac{\boldsymbol{\theta} - \boldsymbol{x}}{||\boldsymbol{x} - \boldsymbol{\theta}||}$$

where $\overset{d}{=}$ denotes equality in distribution.

We now illustrate the use of the R version of the hdepth, (and associated) functions using a real data example. The mrfDepth package includes a subset of the Character Trajectories dataset. This dataset will be used throughout this chapter to illustrate the different functionality of the mrfDepth package This dataset was derived from a bigger found on on the UCI Machine Learning Repository by [Bache and Lichman, 2013, Williams et al., 2006]. The data was processed and interpolated such that observations are obtained for a hundred equally spaced time points. The original data set only contained information on the speed of change in the horizontal and vertical position of the tip, but this data was integrated to obtain information on the position of the horizontal x and vertical y coordinate of the pen. Only the trajectories corresponding to writing the letter 'a' were retained. The data set therefore consists of 171 functional observations resulting in an array of dimensions $100 \times 171 \times 4$. The first two dimensions correspond to the position of the pen whereas the last two variables consist of the speed profiles. Figure 5.4 gives a visual representation of this dataset. In particular, in this section, we will consider the bivariate data matrix comprised of the vertical and horizontal location at time point 35 (so that we have 171 observations recorded on two variables). The code below creates a new data matrix object named X and containing all the observations corresponding to the 35th cross section.

```
R> data(characterTrajectories)
R> X <- characterTrajectories[35 , , ]
```

Next, we can use the maxhdepth function to compute the Tukey median of the dataset X:

```
R> Tukey_med <- maxhdepth(X)
```

In particular, the $med component of the output of the maxhdepth function shows the coordinates of the point having maximal depth (in this instance, $\boldsymbol{z}^+$ is unique) and the $depth component shows the depth of $\boldsymbol{z}^+$ (in this case $\boldsymbol{z}^+$ has depth 0.425.)

```
R> Tukey_med$med
R> -21.75637 -15.98979
R> Tukey_med$depth
R> 0.4444444
```

It is possible to use the Stest and hdepth functions jointly to test the null hypothesis that the distribution of $\boldsymbol{X}$ is angularly symmetric about a point $\boldsymbol{\theta}_0$. In this case, we fix $\boldsymbol{\theta}_0$ to be the Tukey median of $\boldsymbol{X}$:

```
R> HS_depth2 <- hdepth(X , z = Tukey_med$med)
R> Stest(HS_depth2)
```

Here, the `z` argument in `hdepth` specifies the point(s) whose depth should be computed with respect to the data cloud $\boldsymbol{X}$ (declaring the `z` argument is optional. `z` can be either a p-vector or an m by p matrix of coordinates. If left unspecified the `hdepth` function takes `z = X`). The function `Stest` takes as input the result of a call to `hdepth` and, for every p-vector of the argument `z[i ,]` of `hdepth`, returns the p-value of the test that $\boldsymbol{\theta}_0 =$ `z[i ,]` (for example, in this case, the p-value returned by `Stest` for $\boldsymbol{\theta}_0 = \boldsymbol{z}^+$ is ≈ 0.875). Finally, we can illustrate the use of the `isohdepth` function by computing and displaying the depth contours for $\alpha = \{0.125, 0.25, 0.375\}$:

```
R> isohdepth_obj <- isohdepth( X , alpha = c(0.125 , 0.25 ,
                            0.375))
R> plot(isohdepth_obj)
```

The function `isohdepth` computes all the elements needed for producing an isohdepth plot. The actual plotting itself is done by the `plot` functions, which has been overloaded to handle object of class `mrfDepth`. Specifically, when given the result of a call to `isohdepth` as input, the `plot` function returns the plot of the depth contours with depth α (as before α can also be a vector, in which case multiple, nested depth contours will be plotted). In this case, the resulting plot, consisting of a scatterplot of the data as well as the three depth contours is shown in the left plot of Figure 5.1.

The next function we will discuss is the `bagplot`, a wrapper for the algorithm of the same name which was introduced by [Rousseeuw et al., 1999a] to generalize the univariate boxplot to the bivariate setting. The bagplot is constructed using halfspace depth and consists of four main elements. A bag that contains the 50% data points with highest depth, a fence separating the outliers from the inliers as well as a loop indicating the points outside the bag but inside the fence. Finally, the halfspace median is plotted as an estimator for the center of the data.

The bagplot itself is constructed as follows. Firstly the bag is obtained as an interpolation between the biggest depth region that contains less than half of the data points and the smallest depth region that contains at least half of the data points. By inflating the bag from the center outwards with a factor of three, the fence is obtained. Data points outside this fence are flagged as outliers. However, the loop is calculated as the convex hull of the data points inside the fence, i.e., all the data points not flagged as outliers.

It was already possible to calculate and plot the original bagplot using halfspace depth by use of `aplpack` by [Wolf and Bielefeld, 2013] in `R` and the `LIBRA` toolbox in `MATLAB`. However these implementations are made in their native programming language whereas our implementation uses the `Fortran` code by [Rousseeuw et al., 1999a], making it much faster.

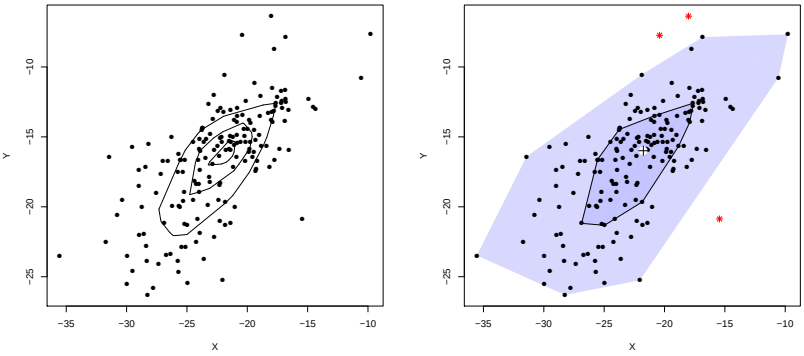


Figure 5.1: (Left) Contours D_α for $\alpha = \{0.125, 0.25, 0.375\}$. (Right) bagplot contours corresponding to the 35th cross section of the Character Trajectories data set.

We now illustrate the use of the `bagplot` function on the same data matrix $\boldsymbol{X}$ corresponding to the 35th cross section of the letter trajectories dataset used above. The routine `bagplot` is used to calculate all the elements of the bagplot. It takes an $n \times 2$ data matrix $\boldsymbol{X}$ as input and returns the Tukey median of $\boldsymbol{X}$ (component `$center` of the output), an integer vector containing the indexes of the eventual observations outside the fence (component `$flag`) and an n by 3 matrix giving for each data point its coordinates and its type (component `$datatype`). This type is determined by the position of the point in the bagplot and indicates whether the data point lies inside the bag (`datatype=1`), the fence (`datatype=2`) or outside the fence (`datatype=3`). Because for very large datasets the computation time of `bagplot` becomes intractable, we also added the possibility to do the main part of the calculations on a random subset of the data (The size of this subset can be controlled by the `sizesubset` option but defaults to 750 points). However, note that using random subsetting also causes the result of the `bagplot` to be no longer deterministic.

```
R> bagplot_output<-bagplot( X , sizesubset = 750)
R> plot(bagplot_output)
```

Again, we have overloaded the `plot` function with specific methods for objects of class `mrfDepth` and entering the above commands results in the plot shown in the right subplot of Figure 5.1.

Next, we illustrate the use of the `R` version of the `outlyingness`, `adjOutlyingness`, `bagplot` (and associated) functions using as before the 35th cross section of the subset of the character trajectory data set corresponding to the letter 'a'. Setting the `ndir` argument to at least $\binom{171}{2}$ (which is the size of $\mathcal{B}_p(\boldsymbol{X})$ in this example) we force the `outlyingness` functions will automatically use all members of $\mathcal{B}_p(\boldsymbol{X})$ in the search for the maximal projection outlyingness index, ensuring that the results are fully deterministic:

```
R> Ndir <- choose(nrow(X) , ncol(X))
R> Outlyingness_output <- outlyingness( X , ndir = Ndir ,
                                    z = NULL , type = "Affine" ,
                                    scaleCenter = "MedMad" , h =
                                     NULL)
```

The `z` argument is left to its default value (`NULL`) so that the code will only compute the outlyingness (and derived measures) corresponding to the p vectors in $\boldsymbol{X}$. The `type` argument determines the set from which the vector of directions are drawn. The default value, `type="Affine"`, draws directions from $\mathcal{B}_p(\boldsymbol{X})$ while `type="Rotation"` uses directions from $\mathcal{B}_2(X)$. The option `scaleCenter` determines the choice of (μ, σ) in Equation (5.1): the default, `scaleCenter="MedMad"` uses $(\text{med}(\boldsymbol{Xa}), \text{mad}(\boldsymbol{Xa}))$ while setting `scaleCenter="unimcd"` uses, $(\mu_{\text{MCD}}(\boldsymbol{Xa}), \sigma_{\text{MCD}}(\boldsymbol{Xa}))$, the univariate MCD estimators of location and scale instead. Finally, when `scaleCenter="unimcd"`, the last argument (`h`) determines the size of the active subset used in the computation of MCD. The `outlyingness` function outputs an n-vector of outlyingness of the vectors in $\boldsymbol{X}$ denoted `$Outlyingness.x` and with values $\{\text{PO}(\boldsymbol{x}_i, \boldsymbol{X})\}_{i=1}^n$ and, when `z` is not set to `NULL`, an m-vector of outlyingness of the vectors in $\boldsymbol{Z} = \{\boldsymbol{z}_i\}_{i=1}^m$ denoted `$Outlyingness.z` with values $\{\text{PO}(\boldsymbol{z}_i, \boldsymbol{X})\}_{i=1}^m$. The `outlyingness` function also outputs a flag `$singularDirections` taking value 1 if during the search procedure, the algorithm encountered a direction $\boldsymbol{a}$ such that $\hat{\sigma}(\boldsymbol{Xa}) = 0$.

The command below illustrates the use of `adjoutlyingness` using the same data set as above, and to make the results more readily comparable, also forcing the algorithm to search for the projection with largest adjusted outlyingness exhaustively, e.g. over all members of $\mathcal{B}_p(\boldsymbol{X})$ (the arguments have the same meaning as their counterparts from the `outlyingness` function):

```
R> adjOutlyingness_output <- adjOutlyingness( X , ndir =
```

```
                                    Ndir , z = NULL , type =
                                      "Affine")
```

The `adjOutlyingness` function also returns a component `$adjout.x` (corresponding to the `$Outlyingness.x` member of the result of a call to the `outlyingness` function) and an optional component `$adjout.z` (corresponding to `outlyingness`'s `$Outlyingness.z`) as well as a flag `$singularDirections` taking value 1 if during the search procedure, the algorithm encountered a direction $\boldsymbol{a}$ for which $\mathrm{IQR}(\boldsymbol{X}\boldsymbol{a}) = 0$. In fact, the outputs from `outlyingness` and `adjOutlyingness` share a unified structure so that companion functions can be easily built that are compatible with both. For example, the `mrfDepth` package includes two such functions, `projCenter` and `projDepth` which we discuss below. All three take as input the result of a call to either the `outlyingness` or `adjOutlyingness` and adapt their behaviour accordingly.

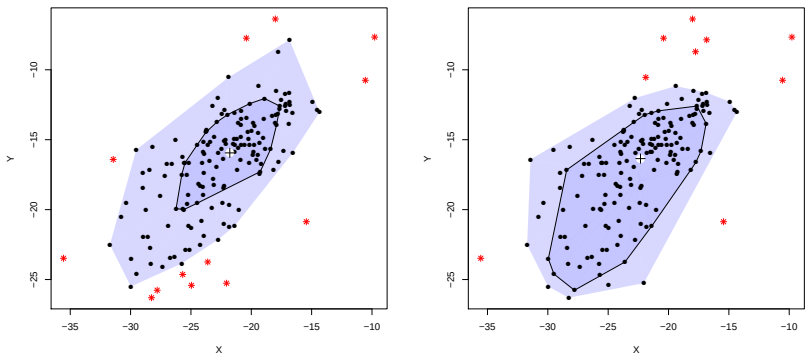


Figure 5.2: Projection Outlyingness (left) and Adjusted Outlyingness (right) contours corresponding to the 35th cross section of the Character Trajectories data set.

When $p = 2$ and the input to `plot` is the result of a call to `outlyingness`, the function will produce a scatter plot showing the original observations as well as the boundary of two convex, nested set. Denoting $\boldsymbol{X}$ The first (innermost) of these two sets (often called the bag) is formed of the convex hull of those observations for which:

$$\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}) \leqslant \operatorname{med} \mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}).$$

The second (and larger) of these two sets (often called the loop) marks the convex hull of those vectors of $\boldsymbol{X}$ for which

$$\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}) \leqslant \sqrt{\chi^2_2(0.99)} \tag{5.11}$$

and observations outside this region are flagged as gross outliers. The `$nonOut.x` component of the output of `outlyingness` returns a boolean indicating whether $\boldsymbol{x}_i$ satisfies Equation (5.11). Likewise, when `z` is not left to the default value `NULL`, the `$nonOut.z` component returns a boolean indicating whether $\mathrm{PO}(\boldsymbol{z}_i, \boldsymbol{X}) \leqslant \sqrt{\chi^2_2(0.99)}$ for each row $\boldsymbol{z}_i$ of $\boldsymbol{Z}$. An example is shown in the left subplot of Figure 5.2 where the bag is shaded in a slightly darker color and the loop in a lighter hue. The gross outliers (corresponding to those observations that are outside the loop) are shown as larger, red, crosses. Similarly to what is done for Projection Outlyingness, when the input to `plot` is the result of a call to `adjOutlyingness`, the smaller of these two regions (the bag) is the convex hull of those observations for which:

$$\mathrm{AO}(\boldsymbol{x}_i, \boldsymbol{X}) \leqslant Q_2(\{\mathrm{AO}(\boldsymbol{x}_i, \boldsymbol{X})\}_{i=1}^n)$$

and the larger region (or loop) is the convex hull of those observations for which:

$$\mathrm{AO}(\boldsymbol{x}_i, \boldsymbol{X}) \leqslant c_2(\{\mathrm{AO}(\boldsymbol{x}_i, \boldsymbol{X})\}_{i=1}^n) \tag{5.12}$$

where c_2 and Q_3 are defined as in Section 5.2 and, again, observations outside this region are flagged as gross outliers. Like the `outlyingness`, the `adjOutlyingness` function also returns an n-vector (resp. m-vector) of booleans `$nonOut.x` (resp. `$nonOut.z`) indicating whether the corresponding row of $\boldsymbol{X}$ (resp. $\boldsymbol{Z}$) has an adjusted outlyingness smaller than $c_2(\{\mathrm{AO}(\boldsymbol{x}_i, \boldsymbol{X})\}_{i=1}^n)$. For example, the code below produces a plot based on the outlyingness (resp. adjusted outlyingness) index produced by the `outlyingness` (resp. `adjOutlyingness`) function:

```
R> plot(Outlyingness_output)
R> plot(adjOutlyingness_output)
```

and the result is shown in Figure 5.2.

The function `projCenter` also takes as input the results of a call to either `outlyingness` or `adjOutlyingness`. In both cases, it returns two estimates of location derived from the given (adjusted) outlyingness index. The first corresponds to the coordinates of the observations having smallest (adjusted)

outlyingness while the second is the mean of the observations lying inside the bag. When fed with the result of a call to `outlyingness`, the function `projCenter` also returns the Huber estimate of location, a weighted mean of the vectors in $\boldsymbol{X}$ where the weight function is the so-called Huber weight function [Maronna and Yohai, 1995]:

$$w(\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X})) = 1\{\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}) \leqslant c\} + 1\{\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}) > c\}(c/\mathrm{PO}(\boldsymbol{x}_i, \boldsymbol{X}))^2$$

with $c = \sqrt{\chi^2_p(0.95)}$. The code below computes the first of these estimates of location using the previously computed result of a call to `outlyingness` (resp. `adjOutlyingness`). The following line of code plots a black star in a white circle on top of the dot corresponding to the observation having smallest outlyingness index, as shown in the left plot of Figure 5.2:

```
R> center_max <- projCenter(Outlyingness_output)$center_max
R> points(t(center_max) , pch = 16 , cex = 2 , col = "white")
R> points(t(center_max) , pch = 3 , cex = 1.5)
```

Replacing the first line of the code above by

```
R> center_max <- projCenter(adjOutlyingness_output)$center_max
```

yields a similar outcome, but this time for the subplot based on adjusted outlyingness and shown in the right plot of Figure 5.2. Compared to the plot produced by `bagplot`, the outlyingness contours derived from the output of the `outlyingness` and `adjOutlyingness` functions (shown in the plots of Figure 5.2 respectively) all three loops are upward slopping, though the one produced by the bagplot is somewhat more drawn towards the data points on the lower left side of the plot window and the one built from the output of `outlyingness` is somewhat tighter than the other two.

5.3 Regression Depth

The `mrfDepth` package also includes wrappers to various algorithms designed to compute the regression depth of an hyperplane with respect to a data cloud $\{(\boldsymbol{x}_i, y_i); i = 1, \ldots, n\} \subset \mathbb{R}^p$. Regression depth can be seen as a generalization of the univariate median to the (multiple) regression setting. First introduced in [Rousseeuw and Hubert, 1999], it is defined as a property of a *fit* (typically indexed by a p-vector of coefficients $\boldsymbol{\theta}$) rather than a property of an observation.

Given an n by p dataset $(\boldsymbol{X}, Y)$, where $Y \in \mathbb{R}^n$ (with $n > p+1$), the depth of any given *candidate* fit $\boldsymbol{\theta}$ with respect to $(\boldsymbol{X}, Y)$, denoted $\text{rdepth}(\boldsymbol{\theta}, \boldsymbol{X}, Y)$, is the smallest number of observations of $(\boldsymbol{X}, Y)$ that would need to be removed in order to make $\boldsymbol{\theta}$ a nonfit, divided by the total size of the dataset. More precisely, denoting $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)^\top$ and the residuals of $(\boldsymbol{x}_i, y_i)$ from the fit $\boldsymbol{\theta}$ as $r_i(\boldsymbol{\theta}) = y_i - \theta_1 - \sum_{j=2}^{p} \theta_j x_{j-1}$, a candidate fit $\boldsymbol{\theta}$ is called a nonfit w.r.t. $(\boldsymbol{X}, Y)$ if and only if there exists an affine hyperplane $\boldsymbol{v}$ in $\boldsymbol{x}$-space such that no $\boldsymbol{x}_i$ belongs to $\boldsymbol{v}$ and such that $\boldsymbol{v}$ separates the observations with positive residuals from the observations having negative residuals. Intuitively, a regression hyperplane is called a nonfit if it can be rotated to vertical (i.e., parallel to the axis of any of the dependent variables) without passing through any data points (the points lying exactly on the hyperplane are counted as "passed through"). In this sense, the deepest regression hyperplane is the most balanced candidate fit $\boldsymbol{\theta}$ in the sense of corresponding to the hyperplane that is most surrounded by the data cloud. In fact, for any dataset $(\boldsymbol{X}, Y)$ in general position in $\mathbb{R}^p$ it holds that:

$$\frac{1}{n}\left\lceil \frac{n}{(p+1)} \right\rceil \leqslant \max_{\boldsymbol{\theta}} \text{rdepth}(\boldsymbol{\theta}, \boldsymbol{X}, Y) \leqslant \left\lceil \frac{n+p}{2} \right\rceil \frac{1}{n} \tag{5.13}$$

Furthermore, for any $(\boldsymbol{x}, y)$-distribution H in $\mathbb{R}^p$ having a density and satisying:

$$\text{med}(y|\boldsymbol{x}) = \tilde{\theta}_1 + \sum_{j=2}^{p} \tilde{\theta}_j x_{j-1} \tag{5.14}$$

for some $\tilde{\boldsymbol{\theta}} = (\tilde{\theta}_1, \ldots, \tilde{\theta}_p) \in \mathbb{R}^p$ then

$$\max_{\boldsymbol{\theta}} \text{rdepth}(\boldsymbol{\theta}, H) = \text{rdepth}(\tilde{\boldsymbol{\theta}}, \boldsymbol{X}, Y) = \frac{1}{2} \tag{5.15}$$

Regression depth has several desirable properties that distinguishes it among the generalization of the median to the regression setting such as the L^1 regression [Koenker, 2005]. Firstly, the functional form of the deepest regression fit is parametric (it assumes the linearity of the conditional median) but places no assumptions on the error distribution. In particular, the model allows for skewed or heteroskedastic disturbances. Secondly, as we illustrate below, the regression depth toolset allows to formerly test the null hypothesis of linearity of the conditional median [Van Aelst et al., 2002b]. Thirdly, the regression depth of a fit with respect to a point cloud $(\boldsymbol{X}, Y)$ only depends on the $\boldsymbol{x}_i$ and the sign of the residuals $r_i(\boldsymbol{\theta})$ (but not their magnitude). This property makes the deepest regression estimator equivariant to monotone transformations of the response. Finally, the deepest regression estimator is very robust. For any $(\boldsymbol{X}, Y)$ sampled

from a distribution H on $\mathbb{R}^p$ with a strictly positive density satisfying Equation (5.14), the finite sample replacement breakdown point of the deepest regression estimator is essentially $\frac{1}{3}$ [Rousseeuw and Hubert, 1999].

The function `maxrdepth` is a wrapper to the various algorithms designed to find the hyperplane $\boldsymbol{\theta}$ having maximal regression depth [Van Aelst et al., 2002b]. The algorithms are exact for dimensions $p \leqslant 2$ and approximate (and random) for higher dimensions. These algorithms were hereto not integrated in a higher level statistical packages such as `R` or `MATLAB` rendering their use by practitioners cumbersome.

We now illustrate the use of the `maxrdepth` function by means of a real data example. We will use a subset of the 'Cars93' dataset [Lock, 1993]. This dataset contains 27 variables describing various characteristics (length, engine type,...) of a sample of 93 cars. The cars were selected randomly by the original authors from among the 1993 passenger car models listed in both the "Consumer Reports" issue and the "Pace Buying Guide". More specifically, for our sample of 93 cars, we are interested in finding the conditional median of "Min Price" (the minimum price for a basic version of the car) as a function of the car's fuel tank capacity, measured in gallons. The code below loads the relevant columns of the `Cars93` dataset (in `R`, the `Cars93` is distributed with the `MASS` library [Venables and Ripley, 2002]). The `maxrdepth` function finds the hyperplane $\boldsymbol{\theta}^*$ having largest depth with respect to the cloud of points $(\boldsymbol{X}, Y)$:

```
R> library("MASS")
R> x <- Cars93[, 17]
R> y <- Cars93[, 4]
R> plot(x, y , pch = 16)
R> reg_depth <- maxrdepth( x , y = y)
```

The `$coef` component of the output of the `maxrdepth` function returns the entries of $\boldsymbol{\theta}^*$ and the `$depth` the depth associated with $\boldsymbol{\theta}^*$.

```
R> reg_depth$depth
R> [1] 0.3870968
```

The following commands compare the regression line having maximum regression depth to the L^1 median line (from the `R` package `quantreg` [Koenker, 2013]) by over-plotting them over the original point cloud:

```
R> library("quantreg")
R> l1_reg <- rq(x[,2] ~ x[,1])
```

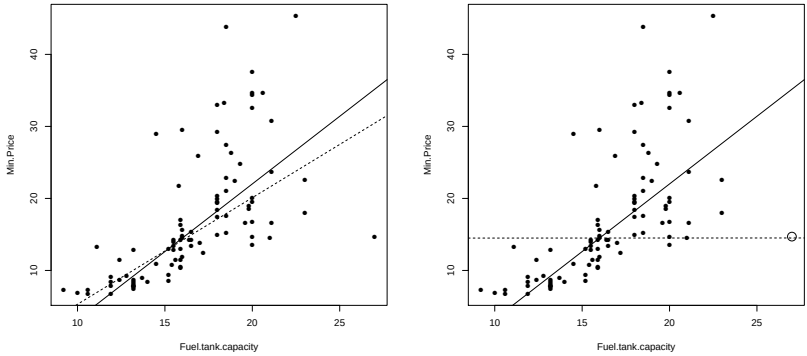


Figure 5.3: The left subplot shows the L^1 (dotted) and maximum depth line for a bivariate data example. The right subplot also shows the L^1 (dotted) and maximum regression depth lines, this time for the situation where the x coordinates of one of the observations has been tempered with.

```
R> abline(l1_reg$coef)
R> abline(reg_depth$coef)
```

The results are shown in the left subplot of Figure 5.3. The left subplot shows the L^1 (dotted) and the line having maximum regression depth. Compared to the L^1 line, the depth based line seems less attracted by the isolated outlier on the lower right corner of the scatter-plot. To illustrate the sturdiness of regression depth to the presence of gross outliers in the data, we multiply the value of the fuel tank capacity corresponding to the isolated outlier on the lower right corner of Figure 5.3 by a factor of 10 (its original location is marked by a white circle on the right subplot of Figure 5.3). We then compute the L^1 and deepest regression for this modified data set again and depict them on the right subplot of Figure 5.3. Compared to the original dataset, the introduction of a single gross outlier has not affected $\boldsymbol{\theta}^*$. However, The L^1 fit obtained on the contaminated dataset is markedly different from the value it had on the original data set.

The `mrfDepth` software package also contains `rdepth`, a wrapper to various functions designed to compute the regression depth of an candidate fit $\boldsymbol{\theta}$ w.r.t. to a sample $(\boldsymbol{X}, Y)$. The algorithms used to compute regression depth have been described in [Rousseeuw and Struyf, 1998]. The calculation is exact for dimensions $p \leqslant 4$ (i.e., a constant carrier and at most three predictors) and approximate for higher dimensions. We will illustrate the use of the `rdepth` function below, using the illustrative example treated above. As we saw earlier,

the depth associated with $\boldsymbol{\theta}^*$ is very close to its upper bound, which is attained at distributions H with linear conditional medians. This observations forms the basis of a test of the null hypothesis that the data set is a sample from a distribution having linear conditional median [Van Aelst et al., 2002b]. The `CMLtest` function implements this test. This function takes as input the result of a call to the `rdepth` function, which computes the regression depth of one (or several) fits `theta` with respect to a data matrix $(\boldsymbol{X}, Y)$ and returns the p-value of the [Van Aelst et al., 2002b] test of linearity of the conditional median. The design of the `CMLtest` function is similar to that of the `Stest` function. In particular, for every p-vector of the argument `theta[i ,]` of `rdepth`, `CMLtest` returns the p-value of the test that $\boldsymbol{\theta}_0$ =`theta[i ,]`. Note that since they are based on sign and ranks, the interpretation of these p-value does not require any parametric assumptions on the distribution of the residuals under the null.

```
R> rdepth_res <- rdepth( x , y = y , theta = reg_depth$coef)
R> CMLtest(rdepth_res)$pval
```

yielding a p-value of 1, so that the sample is not inconsistent with the hypothesis that the population conditional median price of a car is linear in the capacity of the car's fuel tank.

5.4 Functional depth

Formally functional data arrives from variables taking values in a functional space e.g., a Hilbert space. We will restrict ourselves to functional data that can be represented as a vector $\boldsymbol{X}_i(t) = (X^1(t), \ldots, X^p(t))$, with t defined on a closed interval $U \in \mathbb{R}$. The functions $X^j(t)$ with $j = 1, \ldots, p$ take values in $\mathbb{R}$ and it is assumed that $\boldsymbol{X}_i(t) \in C(U)^p$ with $i = 1, \ldots, n$ and N the number of observations. Usually, functional data is only observed on a finite grid of time points $\{t_1, \ldots, t_T\}$. In this case the data can be represented as a three dimensional array having dimensions $T \times N \times p$. For all the functions dealing with functional data, we suppose the data is formatted in this way. To illustrate the functions used in this section, we will use again the Trajectories dataset included in the package.

5.4.1 Multivariate Functional Depth

Multivariate Functional Depth (MFD) combines the local multivariate depths at every time point $t \in U$ defining a global depth for the functional observations.

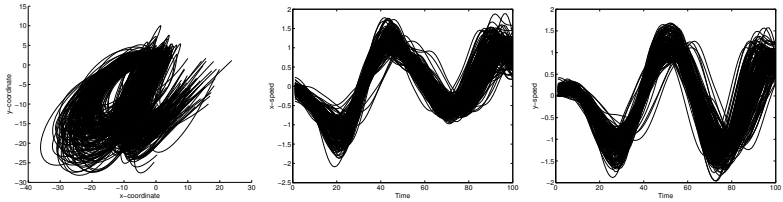


Figure 5.4: Visual representation of the functional data included in the package. Left: the trajectories in the $(x, y)-$plane. Center: the speed profiles of the x-coordinate. Right: The speed profiles of the y-coordinate.

Formally MFD is defined as

$$MFD(X; F_{\mathcal{Y}}) = \int_U D(X(t), F_{\mathcal{Y}(t)}) \cdot w(t)\, dt. \tag{5.16}$$

By incorporating a weight function $w(t)$, multivariate functional depth allows to emphasis certain time regions of the functional observations. Two propositions were made in the original article by [Claeskens et al., 2014]; a uniform weight function or a weight function depending on the volume of the cross-sectional depth contours:

$$w(t) = \frac{vol\left\{D_\alpha(F_{\mathcal{Y}(t)})\right\}}{\int_U vol\left\{D_\alpha(F_{\mathcal{Y}(u)})\right\}\, du}$$

For a functional data sample this definition reduces to

$$MFD_N(X) = \sum_{j=1}^{T} D(X(t_j); F_{\mathcal{Y}(t_j),N}) \ \cdot W_j \tag{5.17}$$

with $W_j = \int_{(t_j+t_{j+1})/2}^{(t_{j-1}+t_j)/2} w(t)\, dt$.

As in the case for multivariate depth functions it is both possible to calculate the depth of a set of functional observations as it is possible to calculate the depth of a set of functional observations to a reference set. Calculation of MFD can be done by use of the `mfd()` function. Several multivariate depth functions can be used for the calculation. By setting the `type = halfspace, projection, adjOutlyingness, simplicial` argument it is possible to choose which depth function is used. The observed time points can be specified with the `time` parameter. By default it is assumed that $t_i = i$. Three standard options are available for the weight function; uniform weights, weights depending on the volume of the cross-sections and finally an option where the vector of weights can be set by the user. Setting the option `alpha` to null will result in uniform weights, setting the parameter `alpha` to a value between 0 and 1 will result

in use of the volume weight function using depth contours of level α. Finally `alpha` can also be equal to a vector of length T.

Two additional parameters are available. Setting `crossdepth` to `1` will return a matrix containing the multivariate depth of the functional observations at each time point. In the case of bivariate functional data, the option `diagnostic=1` will return a logical $N \times T$ matrix signaling whether the multivariate data at each time point is considered to be outlying. In order to detect these local outliers the bagplot routine is used. Therefore this option is not available when the simplicial depth is chosen.

We now illustrate the use of `mfd` function. We first load the data and calculate the multivariate functional depth using the projection depth and the uniform weight function. In the second example we select the first two dimensions corresponding to the horizontal and vertical coordinates of the pen tip. We therefore have bivariate data and are able to ask for the diagnostic option. For illustrative purposes we now use the volume weight function using depth contours of level $\alpha = 0.125$ and also ask to return the cross-sectional depth.

```
R> data( "characterTrajectories" )
R> Trajectories <- characterTrajectories
R> Result <- mfd( Trajectories , type = "projDepth", alpha = 0 )
```

```
R> Data <- characterTrajectories[,,1:2]
R> Result <- mfd( Data , type = "projDepth", crossdepth=TRUE ,
           diagnostic = TRUE )
```

The functional depth median is an estimate of the central tendency of the functional data. [Claeskens et al., 2014] defined the functional depth median by use of the halfspace median at every time point. If the estimate for the center of the multivariate data at time point t is denoted by Θ_t, the functional depth median Θ satisfies $\Theta(t) = \Theta_t$. Therefore the functional depth median does not depend on the weight function and it is typically not one of the observed curves.

The functional depth median can be calculated using the `mfdmedian` routine. Calculation of the `mfdmedian` can be based on the four different depth functions and the choice of depth function is once again controlled by setting the `type` argument. Calculation of the multivariate median is based on the functions `hdepthmedian`, `projmedian` and `aomedian`. Additional parameters to these functions can be passed to the `mfdmedian` routine in a structure via the `options` argument. For simplicial depth only the bivariate maximum depth estimator is available.

We now illustrate this function with two calls. In the first call the halfspace median is used for the cross-sectional estimation. Since the cross-sectional data is four-dimensional, the `maxdir` argument can be set in the underlying `hdepthmedian` routine. In the second call, the projection depth is chosen and the `estimator` argument in the `projmedian` function is changed from its default `maxdepth` to `gravity` (in the latter case, the estimate of location is the center of gravity for the 50 percent points with maximal projection depth).

```
R> Center <- mfdmedian( Trajectories , depthOptions =
                list( maxdir = 50 ) )
R> Center <- mfdmedian( Trajectories , type = "projDepth" ,
                centerOption = "gravity" )
```

5.4.2 Graphical representation of functional data based on depth

Several routines have been implemented to inspect functional data based on multivariate functional depth. The rainbow plot was introduced by [Hyndman and Shang, 2010] and colors the curves according to their depth value. The plot can be constructed using the `frainbowplot` routine. The routine expects as argument the functional observations and their depth. Optional input arguments include a `time` parameter to specify the time vector of observed time points as well as a `col` parameter. With the `col` parameter the user can specify the colors used to make the color scale on the rainbowplot. Interpolation between the provided colors is carried out when less colors are provided than there are functional observations. Finally the `VarLayout` argument can be used to control which variable needs to be plotted on the screen. The input takes the form of a matrix where the subplot at position i, j corresponds to the variable specified at position i, j of the matrix. A zero can be used to signal that the corresponding position in the plot should remain empty. By default only the rainbowplot of the first variable will be plotted.

Once again we illustrate this routine. Naturally the mfd-depths need to be calculated first. Here we use the option `type = 'spDepth'` to compute MFD with the projection depth. In the first example we illustrate the `col` argument by specifying an rgb matrix. In the second call the use of the `VarLayout` argument is illustrated. The resulting plot is shown in Figure 5.5. By using a zero in the `VarLayout` matrix we signal that no plot should be made at that position in the figure.

```
R> Result <- mfd( Trajectories , type = 'spDepth' );
```

```
R> Colors <- matrix( c(0.9 , 0.8 , 0.8 , 0.4 ) , 2 ,
              2 , byrow = TRUE )
R> frainbowplot( Trajectories , depths = Result$MFDdepth ,
              col = Colors )
R> VL <- matrix( c( 1 , 2  ) , 1 , 2 , byrow = TRUE )
R> frainbowplot( Trajectories , depths = Result$MFDdepth ,
              VarLayout = VL )
```

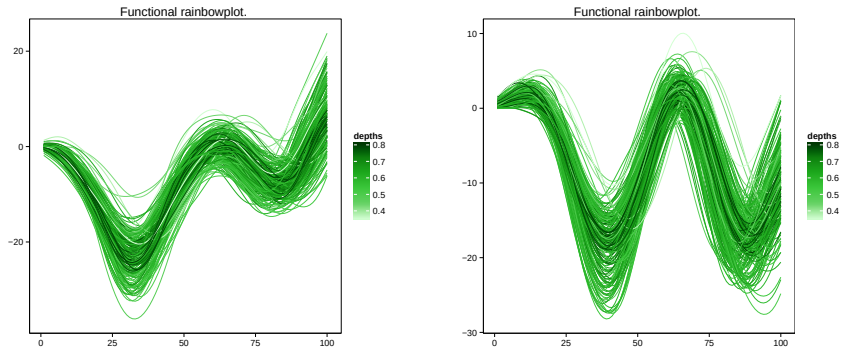


Figure 5.5: Rainbowplot of the Trajectories data set.

A possibility to look at the amount of dispersion in the data is to look at β-central regions of the data. The β-central region is defined as the convex hull off all the graphs of the β-percent functional observations of highest depths. Using the `centralregion` function one can draw central regions with cutoff values specified through the argument `beta`. Required arguments are the data and a vector of percentages. Furthermore it is possible to give extra arguments in the same way as for the `frainbowplot` function. In particular the `time`, `col` and `layout` arguments are available. However here the rows of the `col` matrix correspond to the color of the central regions.

```
R> Colors <- matrix( c( 1, 1 , 0 , 1 , 0.56 , 0 , 1 , 1 , 1 ) ,
             3 , 3 , byrow = TRUE )
R> VL <- matrix( c( 1 , 2 ), 1, 2 )
R> centralregion( Trajectories , depths = Result$MFDdepth ,
     beta = c( 0.75 , 0.5 , 0.25 ), col = Colors , VarLayout =  VL )
```

Figure 5.6 illustrates the β-central regions corresponding to the horizontal and vertical coordinates of the Trajectories datafor $\beta = \{0.25, 0.5, 0.75\}$. Finally for

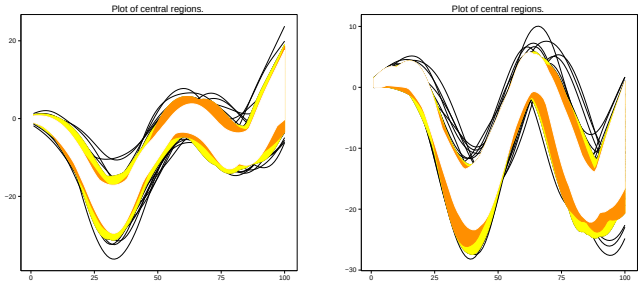


Figure 5.6: The 25%, 50 % and 75% central regions of the horizontal and vertical coordinates of the Trajectories data.

bivariate functional data it is possible to look at local outlyingness based on the diagnostic option from the `mfd` function. If this option is set to `1` an additional logical matrix is returned. The j-th column of this matrix can be used to flag multivariate outliers for time point t_j. If the i-th functional observation has been flagged as a cross-sectional outlier by the bagplot for time point t_j the i, j-th entry of the returned logical matrix will be 1 and 0 otherwise. This allows to make a plot of the data signaling local outlyingness. This plot is obtained by plotting the functional data and coloring all the multivariate points that are considered an outlier.

A final code example shows how this plot can be made. The resulting figure is shown in figure 5.7.

```
R> Result <- mfd( Data , type = 'halfspace' , diagnostic = TRUE )
R> outlyingParts( Data , Result.locOutl )
```

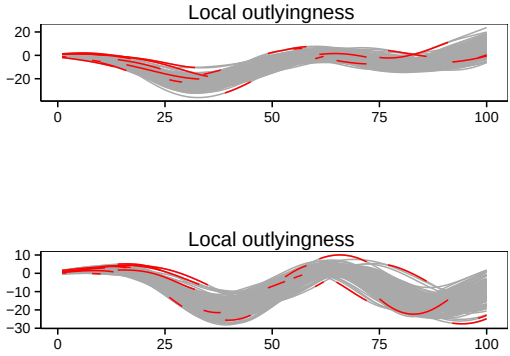


Figure 5.7: Plot of the horizontal and vertical coordinates of the Trajectories data. Observations that cross-sectionally are flagged as outliers by the bagplot are colored red.

Chapter 6

Conclusion

In this thesis, a number of methods were considered, each exploring new terms of compromises between the various requirements that a robust algorithm must satisfy.

In chapter two, through an extensive simulation study, we quantified the robustness of a large panel of state of the art robust estimators of covariance. We Considered in particular the context of one of the most fundamental problems of multivariate robust statistics (robust estimation of location vector and scatter matrix in moderate dimensional data sets in the presence of so-called Tukey-Huber contamination [Tukey, 1962]). There we found that in the presence of outliers, no state of the art robust estimator could systematically find a fit close to the one it would have found without the outliers. Interestingly, we found that the performance of state of the art robust fitting and outliers detection procedures we tried still depends to a large extent on the configuration of the outliers. More precisely, we showed that many situations in which the rate of contamination of the sample is high or the configuration of the outliers are difficult trump all the methods we compared in the sense of rendering them unable to recover the multivariate patterns characterizing the genuine observations in the data. In this chapter we also proposed an innovative approach to comparing the fit of two (or more) robust estimators on real data applications where the identity of the potential outliers is not known. This last idea has been used in a later publication [Schmitt and Vakili, 2015] where it forms the basis of a selection rule allowing to combine two fits while preserving the finite sample breakdown point of the most robust of them. In any case, future area of research of interest could be to extent the comparison carried in this chapter to algorithms designed for robust estimation of outlier detection in

the context of multivariable regression.

Chapter three is devoted to the various DetR algorithms. These are quick, robust and deterministic designed for robust estimation and anomaly detection in the classical multi-variable regression settings. This work is supported by an extensive software package, the DetR package. The main code in this package are written in the C++language using high performance and modern libraries. To ensure ease of use, this software is distributed as a documented and portable multi-platform R package. To facilitate reproducibility and discussion, the package also include the data set and simulation code used to obtain the results shown in the chapter. Finally, in order to facilitate verification of the compliance of the implementation with the proposed algorithm, the package also contains full implementation of the proposed method in the slow but transparent R programing language as well as code to compare the results and intermediate outputs of those code with those of the main C++implementation. Here an obvious subject for further research could be to establish the consistency of the proposed estimators. In Chapter three, we discussed some of the motivating factors that justify the choice for the rather simple design of the proposed algorithm. Of course, depending on the envisioned trade-offs, it can also be possible to design more complicated algorithms for fitting the robust multi-variable linear model deterministically, perhaps by using more deterministic starts as is done for example in DetMCD in the context of robust and deterministic estimation of location vector and scatter matrix.

In chapter four, we focused on the problem of quantifying the performance of a large collection of algorithms for analyzing functional data sets that included MFHD, a new proposal. Through a simulation study, we showed that MFHD was the best amongst state-of-the-arts algorithms for functional data analysis in terms of its ability to withstand some forms of contamination chosen amongst those considered in the literature on the subject. These simulation presage that MFHD should perform quiet well in practice. Furthermore, MFHD could in principle be further extended in various ways, for example to the related problem of classification or clustering of curves. This work naturally lead to greater interest in so called integrated approaches to functional data analysis. Consequently, new approaches have been proposed to improve upon MFHD in important settings such as anomaly detection [Hubert et al., 2015b].

Chapter five described a new software package for `R` and `MATLAB`. It combines existing and newly implemented software for the calculation of depth based estimators and tests for multivariate and functional data as well as for regression. It improves on existing implementations in several ways such as speed but also it gives the user more options, such as allowing for application of depth based approaches to supervised classification tasks which was not possible before. Furthermore the software is implemented in both `R` and `MATLAB` using `Fortran`

and C++code written so as to ensure cross-platform consistency of the results. While the full library has not yet been released for public use, components of it have been used in ongoing research, notably [Hubert et al., 2015a] and [Hubert et al., 2015b] where the huge speed up those components offered over existing implementations allowed for a large increase in the scope of the simulations that could be carried.

Bibliography

[Arrabis-Gil and Romo, 2012] Arrabis-Gil, A. and Romo, J. (2012). Robust depth-based estimation in the time warping model. *Biostatistics*, (13):398–414. pages 86

[Atkinson et al., 2004] Atkinson, A., Riani, M., and Cerioli, A. (2004). *Exploring Multivariate Data With the Forward Search*. Springer. pages 42

[Bache and Lichman, 2013] Bache, K. and Lichman, M. (2013). UCI machine learning repository. pages 97

[Beran, 2003] Beran, R. (2003). Impact of the bootstrap on statistical algorithms and theory. *Statistical Science*, (18):175–184. pages iii, v

[Bernholt, 2006] Bernholt, T. (2006). Robust estimators are hard to compute. Technical report, Universität Dortmund. Technical Report, tr 52–05. pages 5

[Berrendero et al., 2011] Berrendero, J., Justel, A., and Svarc, M. (2011). Principal components for multivariate functional data. *Computational Statistics and Data Analysis*, (55):2619–2634. pages 61

[Billor et al., 2000] Billor, N., Hadi, A. S., and Velleman, P. F. (2000). Bacon: Blocked adaptive computationally efficient outlier nominators. *Computational Statistics and Data Analysis*, (34):279–298. pages 6, 12, 40

[Bremner et al., 2008] Bremner, D., Chen, D., Iacono, J., Langerman, S., and Morin, P. (2008). Output-sensitive algorithms for Tukey depth and related problems. *Statistics and Computing*, (18):259–266. pages 68

[Brinkman, 1981] Brinkman, N. D. (1981). Ethanol fuel - a single-cylinder engine study of efficiency and exhaust emissions. *SAE transactions*, 90:1410–1424. pages 2

[Brys et al., 2005] Brys, G., Hubert, M., and Rousseeuw, P. J. (2005). A robustification of independent component analysis. *Journal of Chemometrics*, 19(5-7):364–375. pages 93

[Brys et al., 2004] Brys, G., Hubert, M., and Struyf, A. (2004). A robust measure of skewness. *Journal of Computational and Graphical Statistics*, 13:996–1017. pages 93

[Campbell et al., 1998] Campbell, N. A., Lopuhaä, H., and Rousseeuw, P. J. (1998). On the calculation of a robust S-estimator of a covariance matrix. *Statistics in Medicine*, (17):2685–2695. pages 36

[Chen and Tyler, 2002] Chen, Z. and Tyler, D. E. (2002). The influence function and maximum bias of Tukey's median. *The Annals of Statistics*, 30:1737–1759. pages 68

[Claeskens et al., 2014] Claeskens, G., Hubert, M., Slaets, L., and Vakili, K. (2014). Multivariate Functional Halfspace depth. *Journal of the American Statistical Association*, 109:411–423. pages 65, 90, 108, 109

[Cuesta-Albertos and Nieto-Reyes, 2008] Cuesta-Albertos, J. A. and Nieto-Reyes, A. (2008). The random Tukey depth. *Computational Statistics and Data Analysis*, 52:4979–4988. pages 68, 96

[Cuevas et al., 2006] Cuevas, A., Febrero, M., and Fraiman, R. (2006). On the use of the bootstrap for estimating functions with functional data. *Computational Statistics and Data Analysis*, (51):1063–1074. pages 82

[Cuevas et al., 2007] Cuevas, A., Febrero, M., and Fraiman, R. (2007). Robust estimation and classification for functional data via projection-based depth notions. *Computational Statistics*, (22):481–496. pages 7, 62, 63, 82, 89

[Daudin et al., 1988] Daudin, J. J., Duby, C., and Trecourt, P. (1988). Stability of principal component analysis studied by the bootstrap method. *Statistics*, (19):241–258. pages 23

[Davies, 1995] Davies, P. L. (1995). Data features. *Statistica Neerlandica*, 49(2):185–245. pages 3

[Davies and Gather, 2005] Davies, P. L. and Gather, U. (2005). Breakdown and groups. *Ann. Statist.*, 33(3):977–1035. pages 4

[De Ketelaere et al., 2011] De Ketelaere, B., Mertens, K., Mathijs, F., Diaza, D., and De Baerdemaeker, J. (2011). Nonstationarity in statistical process control-issues, cases, ideas. *Applied Stochastic Models in Business and Industry*, (27):367–376. pages 7, 62, 69

[Debruyne and Hubert, 2009] Debruyne, M. and Hubert, M. (2009). The influence function of the Stahel-Donoho covariance estimator of smallest outlyingness. *Statistics and Probability Letters*, (79):275–282. pages 20

[Donoho, 1982] Donoho, D. L. (1982). *Breakdown properties of multivariate location estimators.* PhD thesis, Dept. Statistics, Harvard Univ. pages 4, 50, 91

[Donoho and Gasko, 1992] Donoho, D. L. and Gasko, M. (1992). Breakdown properties of location estimates based on halfspace depth and projected outlyingness. *Ann. Statist.*, 20(4):1803–1827. pages 96

[Dwyer, 1967] Dwyer, P. S. (1967). Some applications of matrix derivatives in multivariate analysis. *Journal of the American Statistical Association*, 62(318):pp. 607–625. pages 33

[Febrero et al., 2008] Febrero, M., Galeano, P., and González-Manteiga, W. (2008). Outlier detection in functional data by depth measures, with application to identify abnormal NO_x levels. *Environmetrics*, (19):331–345. pages 74

[Febrero-Bande and Oviedo de la Fuente, 2012] Febrero-Bande, M. and Oviedo de la Fuente, M. (2012). Statistical computing in functional data analysis: The r package fda.usc. *Journal of Statistical Software*, (51):1–28. pages 82, 90

[Ferraty and Vieu, 2006] Ferraty, F. and Vieu, P. (2006). *Nonparametric Functional Data Analysis.* Springer Series in Statistics. Springer. pages 7, 61, 89

[Fisk, 2005] Fisk, S. (2005). A very short proof of cauchy's interlace theorem for eigenvalues of hermitian matrices. Technical report, BOWDOIN COLLEGE. arXiv:math/0502408. pages 56

[Fraiman and Muniz, 2001] Fraiman, R. and Muniz, G. (2001). Trimmed means for functional data. *Test*, 10:419–440. pages 7, 63, 89

[Genest et al., 2012] Genest, M., Masse, J.-C., and J.-F., P. (2012). *depth: Depth functions tools for multivariate analysis.* pages 8, 90, 95

[Ginsberg et al., 2009] Ginsberg, J., Mohebbi, M. H., Patel, R. S., Brammer, L., Smolinski, M. S., and Brilliant, L. (2009). Detecting influenza epidemics using search engine query data. *Nature*, (457):1012–1014. pages 23

[Gnanadesikan and Kettenring, 1972] Gnanadesikan, R. and Kettenring, J. (1972). Robust estimates, residuals, and outlier detection with multiresponse data. *Biometrics*, (28):81–124. pages 12, 32

[Guennebaud et al., 2013] Guennebaud, G., Jacob, B., et al. (2013). Eigen v3.2.2. `http://eigen.tuxfamily.org`. pages 8, 39, 92

[Hallin et al., 2010] Hallin, M., Paindaveine, D., and Siman, M. (2010). Multivariate quantiles and multiple-output regression quantiles: From l_1 optimization to halfspace depth. *The Annals of Statistics*, (38):635–669. pages 68

[Huber, 1981] Huber, P. J. (1981). *Robust Statistics*. Wiley. pages 12, 32

[Hubert et al., 2015a] Hubert, M., Reynkens, T., Schmitt, E., and Verdonck, T. (2015a). Sparse PCA for high-dimensional data with outliers. *Technometrics*. pages 117

[Hubert et al., 2015b] Hubert, M., Rousseeuw, P. J., and Segaert, P. (2015b). Multivariate functional outlier detection. *Statistical Methods and Applications*, pages 1–26. pages 116, 117

[Hubert et al., 2008] Hubert, M., Rousseeuw, P. J., and Van Aelst, S. (2008). High-breakdown robust multivariate methods. *Statist. Sci.*, 23(1):92–119. pages 4

[Hubert et al., 2005] Hubert, M., Rousseeuw, P. J., and Vanden Branden, K. (2005). ROBPCA: A new approach to robust principal components analysis. *Technometrics*, 47:64–79. pages 20, 92

[Hubert et al., 2015c] Hubert, M., Rousseeuw, P. J., Vanpaemel, D., and Verdonck, T. (2015c). The DetS and DetMM estimators for multivariate location and scatter. *Computational Statistics and Data Analysis*, (81):64–75. pages 6, 30, 45, 51, 52

[Hubert et al., 2012] Hubert, M., Rousseeuw, P. J., and Verdonck, T. (2012). A deterministic algorithm for robust location and scatter. *Journal of Computational and Graphical Statistics*, (21):618–637. pages 6, 12, 13, 30, 45, 47, 51, 52

[Hubert and Van der Veeken, 2008] Hubert, M. and Van der Veeken, S. (2008). Outlier detection for skewed data. *Journal of Chemometrics*, 22:235–246. pages 89, 93

[Hubert and Vandervieren, 2008] Hubert, M. and Vandervieren, E. (2008). An adjusted boxplot for skewed distributions. *Computational Statistics and Data Analysis*, (52):5186–5201. pages 18

[Hyndman and Shang, 2010] Hyndman, R. J. and Shang, H. L. (2010). Rainbow plots, bagplots, and boxplots for functional data. *Journal of Computational and Graphical Statistics*, 19:29–45. pages 74, 110

[Ieva et al., 2012] Ieva, F., Paganoni, A. M., Pigoli, D., and Vitelli, V. (2012). Multivariate functional clustering for the morphological analysis of electrocardiograph curves. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, pages 401–418. pages 89

[Koenker, 2005] Koenker, R. (2005). *Quantile Regression*. Cambridge University Press. pages 104

[Koenker, 2013] Koenker, R. (2013). *quantreg: Quantile Regression*. R package version 5.05. pages 105

[Lehmann and Casella, 2003] Lehmann, E. L. and Casella, G. (2003). *Theory of Point Estimation*. Springer. pages 52

[Liu et al., 2006] Liu, R., Serfling, R., and Souvaine, D. (2006). *Data depth: robust multivariate analysis, computational geometry and applications*. DIMACS Ser. Discrete Math. Theoret. Comput. Sci. American Mathematical Society. pages 7

[Liu, 1988] Liu, R. Y. (1988). On a notion of simplicial depth. *Proceedings of the National Academy of Sciences of the United States of America*, 85:1732–4. pages 89, 95, 96

[Liu, 1990] Liu, R. Y. (1990). On a notion of data depth based on random simplices. *The Annals of Statistics*, 18:405–414. pages 82, 89

[Lock, 1993] Lock, R. H. (1993). 1993 new car data. *Journal of Statistics Education*, (1). pages 105

[López-Pintado and Romo, 2009] López-Pintado, S. and Romo, J. (2009). On the concept of depth for functional data. *Journal of the American Statistical Association*, 104:718–734. pages 7, 63, 67, 82, 90

[Lopez-Pintado and Romo, 2011] Lopez-Pintado, S. and Romo, J. (2011). A half-region depth for functional data. *Computational Statistics and Data Analysis*, (55):1679–1695. pages 7, 63, 90

[López-Pintado et al., 2010] López-Pintado, S., Romo, J., and Torrente, A. (2010). Robust depth-based tools for the analysis of gene expression data. *Biostatistics*, (11):254–264. pages 67

[Lopez-Pintado and Torrente, 2013] Lopez-Pintado, S. and Torrente, A. (2013). *depthTools: Depth Tools Package*. pages 90

[Lopuhaä and Rousseeuw, 1991] Lopuhaä, H. P. and Rousseeuw, P. J. (1991). Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, (19):229–248. pages 36, 55, 59

[Maronna et al., 2006] Maronna, R. A., Martin, R. D., and Yohai, V. J. (2006). *Robust Statistics: Theory and Methods*. Wiley. pages 6, 11, 12, 37, 51

[Maronna and Yohai, 1990] Maronna, R. A. and Yohai, V. J. (1990). The maximum bias of robust covariances. *Communications Statistics Theory Methods*, (19):3925–3933. pages 14

[Maronna and Yohai, 1995] Maronna, R. A. and Yohai, V. J. (1995). The behavior of the Stahel-Donoho robust multivariate estimator. *Journal of the American Statistical Association*, 90:330–341. pages 6, 12, 92, 103

[Maronna and Zamar, 2002] Maronna, R. A. and Zamar, R. H. (2002). Robust estimates of location and dispersion for high-dimensional data sets. *Technometrics*, (44):307–317. pages 6, 12, 18, 32, 40, 47, 48, 53

[MATLAB, 2014] MATLAB (2014). *version 8.2.0.701 (R2013b)*. The MathWorks Inc., Natick, Massachusetts. pages 8, 10, 90

[Mosler, 2013] Mosler, K. (2013). Depth statistics. In Becker, C., Fried, R., and Kuhnt, S., editors, *Robustness and Complex Data Structures*, pages 17–34. Springer Berlin Heidelberg. pages 88

[Ogden et al., 2002] Ogden, R., Miller, C. E., Takezawa, K., and Ninomiya, S. (2002). Functional regression in crop lodging assessment with digital images. *Journal of Agricultural, Biological, and Environmental Statistics*, 7(3):389–402. pages 89

[O'Hanian and Ruffini, 1980] O'Hanian, H. C. and Ruffini, R. (1980). *Gravitation and Spacetime*. W. W. Norton, 2nd edition. pages 4

[Pfeiffer et al., 2002] Pfeiffer, R. M., Bura, E., Smith, A., and Rutter, J. L. (2002). Two approaches to mutation detection based on functional data. *Statistics in Medicine*, 21(22):3447–3464. pages 89

[Pigoli and Sangalli, 2012] Pigoli, D. and Sangalli, L. (2012). Estimation of multidimensional curves and their derivatives. *Computational Statistics and Data Analysis*, (56):1482–1498. pages 61

[Pison et al., 2002] Pison, G., Van Aelst, S., and Willems, G. (2002). Small sample corrections for LTS and MCD. *Metrika*, (55):111–123. pages 38

[R Core Team, 2014] R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. pages 8, 10, 13, 90

[Ramsay and Silverman, 2002] Ramsay, J. and Silverman, B. W. (2002). *Applied Functional Data Analysis*. Springer series in Statistics. Springer. pages 89

[Ramsay and Silverman, 2005] Ramsay, J. and Silverman, B. W. (2005). *Functional Data Analysis*. Springer, 2nd edition. pages 7, 61, 62, 89

[Rocke, 1998] Rocke, D. M. (1998). Constructive statistics: Estimators, algorithms, and asymptotics. Technical report, Center for Image Processing and Integrated Computing. University of California, Davis. pages 5

[Rocke and Woodruff, 1996] Rocke, D. M. and Woodruff, D. L. (1996). Identification of outliers in multivariate data. *Journal of the American Statistical Association*, (91):1047–1061. pages 14

[Romanazzi, 2001] Romanazzi, M. (2001). Influence function of halfspace depth. *Journal of Multivariate Analysis*, 77(1):138 – 161. pages 68

[Rosenberger and Gasko, 1983] Rosenberger, J. and Gasko, M. (1983). Comparing location estimators: Trimmed means, medians, and trimean. In Hoaglin, D., Mosteller, F., and Tukey, J. W., editors, *Understanding Robust and Exploratory Data Analysis*, page 297. Wiley, New York. pages 42

[Rousseeuw, 1984] Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association*, (79):871–880. pages 6, 12, 29, 30, 33, 34

[Rousseeuw and Croux, 1993] Rousseeuw, P. J. and Croux, C. (1993). Alternatives to the median absolute deviation. *Journal of the American Statistical Association*, 88:1273–1283. pages 33, 47

[Rousseeuw et al., 2014] Rousseeuw, P. J., Croux, C., Todorov, V., Ruckstuhl, A., Salibian-Barrera, M., Verbeke, T., Koller, M., and Maechler, M. (2014). *robustbase: Basic Robust Statistics*. pages 39, 40, 46, 47, 49

[Rousseeuw and Hubert, 1999] Rousseeuw, P. J. and Hubert, M. (1999). Regression depth. *Journal of the American Statistical Association*, 94:388–402. pages 8, 89, 103, 105

[Rousseeuw and Leroy, 1987] Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. Wiley. pages 2, 11, 30, 35, 37, 51, 91

[Rousseeuw and Ruts, 1996] Rousseeuw, P. J. and Ruts, I. (1996). Algorithm AS 307: Bivariate Location Depth. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 45:pp. 516–526. pages 68, 95

[Rousseeuw and Ruts, 1998] Rousseeuw, P. J. and Ruts, I. (1998). Constructing the bivariate Tukey median. *Statistica Sinica*, 8:828–839. pages 68, 96

[Rousseeuw and Ruts, 1999] Rousseeuw, P. J. and Ruts, I. (1999). The depth function of a population distribution. *Metrika*, (49):213–244. pages 69

[Rousseeuw et al., 1999a] Rousseeuw, P. J., Ruts, I., and Tukey, J. W. (1999a). The Bagplot: A Bivariate Boxplot. *The American Statistician*, 53:382–387. pages 68, 76, 96, 98, 99

[Rousseeuw and Struyf, 1998] Rousseeuw, P. J. and Struyf, A. (1998). Computing location depth and regression depth in higher dimensions. *Statistics and Computing*, 8:193–203. pages 68, 95, 96, 106

[Rousseeuw and Struyf, 2002] Rousseeuw, P. J. and Struyf, A. (2002). A depth test for symmetry. In Huber-Carol, C., Balakrishnan, N., Nikulin, M., and Mesbah, M., editors, *Goodness-of-Fit Tests and Model Validity*, Statistics for Industry and Technology, pages 401–412. Birkhäuser Boston. pages 8, 96

[Rousseeuw et al., 1999b] Rousseeuw, P. J., Van Aelst, S., and Hubert, M. (1999b). Regression depth: Rejoinder. *Journal of the American Statistical Association*, 94(446):pp. 419–433. pages 41

[Rousseeuw et al., 2004] Rousseeuw, P. J., Van Aelst, S., Van Driessen, K., and Agulló, J. (2004). Robust multivariate regression. *Technometrics*, (46):293–305. pages 23

[Rousseeuw and Van Driessen, 1999] Rousseeuw, P. J. and Van Driessen, K. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, (41):212–223. pages 6, 12, 23, 30, 33, 55

[Rousseeuw and Van Driessen, 2006] Rousseeuw, P. J. and Van Driessen, K. (2006). Computing LTS regression for large data sets. *Data Mining and Knowledge Discovery archive*, (12):29–45. pages 6, 30, 34, 35

[Rousseeuw and Van Zomeren, 1990] Rousseeuw, P. J. and Van Zomeren, B. C. (1990). Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, 85:633–639. pages 3, 43, 49, 50

[Rousseeuw and Yohai, 1984] Rousseeuw, P. J. and Yohai, V. J. (1984). Robust regression by means of s-estimators. In *Robust and Nonlinear Time Series Analysis*, volume 26 of *Lecture Notes in Statistics*, pages 256–272. pages 6, 29, 35

[Ruts and Rousseeuw, 1996] Ruts, I. and Rousseeuw, P. J. (1996). Computing depth contours of bivariate point clouds. *Computational Statistics and Data Analysis*, 23:153–168. pages 96

[Salibian-Barrera et al., 2006] Salibian-Barrera, M., Van Aelst, S., and Willems, G. (2006). PCA based on multivariate MM-estimators with fast and robust bootstrap. *Journal of the American Statistical Association*, (101):1198–1211. pages 12, 24

[Salibian-Barrera and Yohai, 2006] Salibian-Barrera, M. and Yohai, V. J. (2006). A fast algorithm for S-regression estimates. *Journal of Computational and Graphical Statistics*, (15):414–427. pages 6, 12, 30, 35, 36, 39

[Sangalli et al., 2009] Sangalli, L., Secchi, P., Vantini, S., and Veneziani, A. (2009). A case study in exploratory functional data analysis: Geometrical features of the internal carotid artery. *Journal of the American Statistical Association*, (104):2619–2634. pages 61

[Schmitt and Vakili, 2015] Schmitt, E. and Vakili, K. (2015). The fasthcs algorithm for robust pca. *Statistics and Computing*. pages 115

[Seber, 2008] Seber, G. A. F. (2008). *Matrix Handbook for Statisticians*. Wiley. pages 54

[Slaets et al., 2012] Slaets, L., Claeskens, G., and Hubert, M. (2012). Phase and amplitude-based clustering for functional data. *Computational Statistics and Data Analysis*, (56):2360–2374. pages 62

[Stahel, 1981] Stahel, W. A. (1981). *Breakdown of Covariance Estimators*. PhD thesis, ETH Zürich, Zürich. pages 91

[Stahel and Maechler, 2009] Stahel, W. A. and Maechler, M. (2009). *robustX: eXperimental eXtraneous eXtraordinary ... Functionality for Robust Statistics*. pages 13, 15

[Stigler, 1973] Stigler, S. M. (1973). Simon Newcomb, Percy Daniell, and the history of robust estimation 1885-1920. *Journal of the American Statistical Association*, (68):872–879. pages 1

[Sun and Genton, 2011] Sun, Y. and Genton, M. (2011). Functional boxplots. *Journal of Computational and Graphical Statistics*, (20):316–334. pages 67, 70

[Sun and Genton, 2012] Sun, Y. and Genton, M. G. (2012). Adjusted functional boxplots for spatio-temporal data visualization and outlier detection. *Journal of Computational and Graphical Statistics*, (22):54–64. pages 67, 77

[Todorov and Filzmoser, 2009] Todorov, V. and Filzmoser, P. (2009). An object-oriented framework for robust multivariate analysis. *Journal of Statistical Software*, 32:1–47. pages 13, 30, 90, 92

[Trends, 2012] Trends, G. F. (2012). Google flu trends. pages 23

[Tukey, 1962] Tukey, J. W. (1962). The future of data analysis. *Ann. Math. Statist.*, 33:1–67. pages 3, 8, 115

[Tukey, 1975] Tukey, J. W. (1975). Mathematics and the Picturing of Data. *Proceedings of the International Congress of Mathematicians*, volume 2. pages 63, 68, 89, 95

[Tyler, 1994] Tyler, D. E. (1994). Finite sample breakdown points of projection based multivariate location and scatter statistics. *The Annals of Statistics*, 22:1024–1044. pages 51

[Vakili et al., 2012] Vakili, K., Hubert, M., and Rousseeuw, P. J. (2012). The mcs estimator of location and scatter. *Proceedings of the twentieth international conference on Computational Statistics*, pages 825–834. pages 33

[Van Aelst et al., 2002a] Van Aelst, S., Rousseeuw, P. J., Hubert, M., and Struyf, A. (2002a). The deepest regression method. *Journal of Multivariate Analysis*, 81(1):138 – 166. pages 8

[Van Aelst et al., 2002b] Van Aelst, S., Rousseeuw, P. J., Hubert, M., and Struyf, A. (2002b). The deepest regression method. *Journal of Multivariate Analysis*, 81:138–166. pages 104, 105, 107

[Venables and Ripley, 2002] Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition. ISBN 0-387-95457-0. pages 105

[Verboven and Hubert, 2005] Verboven, S. and Hubert, M. (2005). LIBRA: a MATLAB library for robust analysis. *Chemometrics and Intelligent Laboratory Systems*, 75(2):127–136. pages 90

[Verboven and Hubert, 2010] Verboven, S. and Hubert, M. (2010). LIBRA: a MATLAB library for robust analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2:509–515. pages 8

[Wang et al., 2013] Wang, J., Zamar, R. H., Marazzi, A., Yohai, V. J., Salibian-Barrera, M., Maronna, R. A., Zivot, E., Rocke, D., Martin, D., Maechler, M., and Konis, K. (2013). *robust: Robust Library*. pages 90

[Williams et al., 2006] Williams, B. H., Toussaint, M., and Storkey, A. J. (2006). Extracting motion primitives from natural handwriting data. *Artificial Neural Networks : ICANN*, pages 634–643. pages 97

[Wolf and Bielefeld, 2014] Wolf, H. P. and Bielefeld, U. (2014). *aplpack: Another Plot PACKage: stem.leaf, bagplot, faces, spin3R, plotsummary, plothulls, and some slider functions*. R package version 1.3.0. pages 8

[Wolf and Bielefeld, 2013] Wolf, P. and Bielefeld, U. (2013). *aplpack: Another Plot PACKage: stem.leaf, bagplot, faces, spin3R, plotsummary, plothulls, and some slider functions*. pages 90, 99

[Yohai, 1987] Yohai, V. J. (1987). High breakdown-point and high efficiency estimates for regression. *The Annals of Statistics*, (15):642–656. pages 6, 29, 37

[Yohai and Zamar, 1988] Yohai, V. J. and Zamar, R. H. (1988). High breakdown point estimates of regression by means of the minimization of an efficient scale. *Journal of the American Statistical Association*, (83):406–413. pages 33

[Zuo and Serfling, 2000] Zuo, Y. and Serfling, R. J. (2000). General notions of statistical depth functions. *The Annals of Statistics*, 28:461–482. pages 64, 65, 66, 86, 89, 94

List of publications

Articles in internationally reviewed academic journals.

Schmitt, E., V., Vakili, K. (2015). The FastHCS Algorithm for Robust PCA. Statistics and Computing.

Schmitt, E., Öllerer, V., Vakili, K. (2014). The Finite Sample Breakdown Point of PCS. Statistics and Probability Letters, 94, 214-220.

Hubert, M., Rousseeuw, P., Vakili, K. (2014). Shape bias of robust covariance estimators: an empirical study. Statistical Papers, 55 (1), 15-28.

Claeskens, G., Hubert, M., Slaets, L., Vakili, K. (2014). Multivariate functional halfspace depth. Journal of the American Statistical Association, 109 (505), 411-423.

Vakili, K., Schmitt, E. (2014). Finding multivariate outliers with FastPCS. Computational Statistics and Data Analysis, 69 (1), art.nr. 4, 54-66.

Papers at international scientific conferences and symposia, published in full in proceedings.

Vakili, K., Hubert, M., Rousseeuw, P. (2012). The MCS estimator of location and scatter. In Colubi, A. (Ed.), Fokianos, K. (Ed.), Gonzalez-Rodriguez, G. (Ed.), Kontoghioghes, E. (Ed.), Proceedings of Compstat 2012 - 20th International Conference on Computational Statistics. International Conference on Computational Statistics (Compstat 2012). Limassol (Cyprus), 27-31 August 2012 (pp. 825-834).

Hubert, M., Claeskens, G., De Ketelaere, B., Vakili, K. (2012). A new depth-based approach for detecting outlying curves. In Colubi, A. (Ed.), Fokianos, K. (Ed.), Gonzalez-Rodriguez, G. (Ed.), Kontoghiorghes, E. (Ed.), Proceedings of Compstat 2012 - 20th International Conference on Computational Statistics. International Conference on Computational Statistics (Compstat 2012). Limassol

(Cyprus), 27-31 August 2012 (pp. 329-340) The International Statistical Institute/International Association for Statistical Computing.

Meeting abstracts, presented at international scientific conferences and symposia, published or not published in proceedings or journals.

Vakili, K., Hubert, M., Rousseeuw, P., Verdonck, T. (2013). A deterministic algorithm for LTS. ERCIM. London, 14-16 December 2013.

Vakili, K., Hubert, M., Claeskens, G. (2012). Detecting outlying curves based on functional depth. ERCIM 2012. Oviedo (Spain), 1-3 December 2012.

Vakili, K., Hubert, M., Rousseeuw, P. (2011). Comparison of various high-breakdown starting points for robust multivariate location and scatter estimators. ICORS. Valladolid, 27 June - 1 July 2011.

Vakili, K., Hubert, M., Rousseeuw, P. (2012). The MCS estimator of location and scatter. Annual Meeting of the Belgian Statistical Society. Liege, 25-26 October 2012.

FACULTY OF SCIENCE
DEPARTMENT OF MATHEMATICS
STATISTICS SECTION
Celestijnenlaan 200B box 2400
B-3001 Heverlee